A FINITE-ELEMENT SIMULATION MODEL FOR SATURATED-UNSATURATED, FLUID-DENSITY-DEPENDENT GROUND-WATER FLOW WITH ENERGY TRANSPORT OR CHEMICALLY-REACTIVE SINGLE-SPECIES SOLUTE TRANSPORT

By Clifford I. Voss

U.S. GEOLOGICAL SURVEY
Water-Resources Investigations Report 84—4369
Prepared in Cooperation with
U.S. AIR FORCE ENGINEERING AND SERVICES CENTER
Tyndall A.F.B., Florida

1984

UNITED STATES DEPARTMENT OF THE INTERIOR
WILLIAM P. CLARK, Secretary

GEOLOGICAL SURVEY
Dallas L. Peck, Director

For additional information
write to:

Chief Hydrologist
U.S. Geological Survey
431 National Center
Reston, Virginia 22092

Copies of this report can be
purchased from:

U.S. Geological Survey
Open-File Services Section
Western Distribution Branch
Box 25425, Federal Center
Denver, Colorado 80225

11. A Finite-Element Simulation Model for Saturated-Unsaturated, Fluid-Density-Dependent Ground-Water Flow with Energy Transport or Chemically-Reactive Single Species Solute Transport. (UNCLASSIFIED)

19.
SUTRA flow simulation may be employed for areal and cross-sectional modeling of saturated ground-water flow systems, and for cross-sectional modeling of unsaturated zone flow. Solute transport simulation using SUTRA may be employed to model natural or man-induced chemical species transport including processes of solute sorption, production and decay, and may be applied to analyze ground-water contaminant transport problems and aquifer restoration designs. In addition, solute transport simulation with SUTRA may be used for modeling of variable density leachate movement, and for cross-sectional modeling of salt-water intrusion in aquifers at near-well or regional scales, with either dispersed or relatively sharp transition zones between fresh water and salt water. SUTRA energy transport simulation may be employed to model thermal regimes in aquifers, subsurface heat conduction, aquifer thermal energy storage systems, geothermal reservoirs, thermal pollution of aquifers, and natural hydrogeological convection systems.

Mesh construction is quite flexible for arbitrary geometries employing quadrilateral finite elements in Cartesian or radial-cylindrical coordinate systems. The mesh may be coarsened employing 'pinch nodes' in areas where transport is unimportant. Permeabilities may be anisotropic and may vary both in direction and magnitude throughout the system as may most other aquifer and fluid properties. Boundary conditions, sources and sinks may be time-dependent. A number of input data checks are made in order to verify the input data set. An option is available for storing the intermediate results and restarting simulation at the intermediate time. An option to plot results produces output which may be contoured directly on the printer paper. Options are also available to print fluid velocities in the system, and to make temporal observations at points in the system.

Both the mathematical basis for SUTRA and the program structure are highly general, and are modularized to allow for straightforward addition of new methods or processes to the simulation. The FORTRAN-77 coding stressed clarity and modularity rather than efficiency, providing easy access for eventual modifications.

18.
DESCRIPTORS: Two Dimensional Flow, Decay, Adsorption

IDENTIFIERS: Thermal Pollution, Water Pollution, Leaching, SUTRA (Saturated-Unsaturated Transport)

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) Water-Resources Investigations Report 84-4369	5. MONITORING ORGANIZATION REPORT NUMBER(S) ESL-TR-85-10
6a. NAME OF PERFORMING ORGANIZATION U.S. Geological Survey 6b. OFFICE SYMBOL (*If applicable*)	7a. NAME OF MONITORING ORGANIZATION HQ AFESC/RDVW
6c. ADDRESS (*City, State and ZIP Code*) 431 National Center Reston, Virginia 22092	7b. ADDRESS (*City, State and ZIP Code*) Tyndall AFB, Florida 32403
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Jointly funded and sponsored by 6 & 7 above. 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MIPR-N-83-18
8c. ADDRESS (*City, State and ZIP Code*)	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
63723F	2103	90	25

11. TITLE (*Include Security Classification*)

12. PERSONAL AUTHOR(S) Voss, Clifford I.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Yr., Mo., Day*)	15. PAGE COUNT
Final	FROM 821229 TO 850130	841230	409

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)		
FIELD	GROUP	SUB. GR.			
08	08		Ground Water	Transport	Energy
12	01		Mathematical Models	Flow	Fluid Flow
			Computer Programs	Solutes	Radial Flow

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

SUTRA (Saturated-Unsaturated Transport) is a computer program which simulates fluid movement and the transport of either energy or dissolved substances in a subsurface environment. The model employs a two-dimensional hybrid finite-element and integrated-finite-difference method to approximate the governing equations that describe the two interdependent processes that are simulated by SUTRA:

1. fluid density-dependent saturated or unsaturated ground-water flow, and either
2a. transport of a solute in the ground water, in which the solute may be subject to: equilibrium adsorption on the porous matrix, and both first-order and zero-order production or decay, or,
2b. transport of thermal energy in the ground water and solid matrix of the aquifer.

SUTRA provides, as the primary calculated result, fluid pressures and either solute concentrations or temperatures, as they vary with time, everywhere in the simulated subsurface system. SUTRA may also be used to simulate simpler subsets of the above process.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (*Include Area Code*)	22c. OFFICE SYMBOL
1Lt Edward Heyse	(904) 283-4628	HQ AFESC/RDVW

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

PREFACE

This report describes a complex computer model for analysis of fluid flow and solute or energy transport in subsurface systems. The user is cautioned that while the model will accurately reproduce the physics of flow and transport when used with proper discretization, it will give meaningful results only for well-posed problems based on sufficient supporting data.

The user is requested to kindly notify the originating office of any errors found in this report or in the computer program. Updates will occasionally be made to both the report and the computer program to include corrections of errors, addition of processes which may be simulated, and changes in numerical algorithms. Users who wish to be added to the mailing list for updates may send a request to the originating office at the following address:

Chief Hydrologist - SUTRA
U.S. Geological Survey
431 National Center
Reston, VA 22092

Copies of the computer program on tape are available at cost of processing from:

U.S. Geological Survey
WATSTORE Program Office
437 National Center
Reston, VA 22092
Telephone: 703-860-6871

This report has been reviewed by the Public Affairs Office (AFESC/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nationals.

(p.vii follows)

ABSTRACT

SUTRA (Saturated-Unsaturated Transport) is a computer program which simulates fluid movement and the transport of either energy or dissolved substances in a subsurface environment. The model employs a two-dimensional hybrid finite-element and integrated-finite-difference method to approximate the governing equations that describe the two interdependent processes that are simulated:

1) fluid density-dependent saturated or unsaturated ground-water flow, and either

2a) transport of a solute in the ground water, in which the solute may be subject to: equilibrium adsorption on the porous matrix, and both first-order and zero-order production or decay,

or,

2b) transport of thermal energy in the ground water and solid matrix of the aquifer.

SUTRA provides, as the primary calculated result, fluid pressures and either solute concentrations or temperatures, as they vary with time, everywhere in the simulated subsurface system. SUTRA may also be used to simulate simpler subsets of the above process.

SUTRA flow simulation may be employed for areal and cross-sectional modeling of saturated ground-water flow systems, and for cross-sectional modeling of unsaturated zone flow. Solute transport simulation using SUTRA may be employed to model natural or man-induced chemical species transport including processes of solute sorption, production and decay, and may be applied

to analyze ground-water contaminant transport problems and aquifer restoration designs. In addition, solute transport simulation with SUTRA may be used for modeling of variable density leachate movement, and for cross-sectional modeling of salt-water intrusion in aquifers at near-well or regional scales, with either dispersed or relatively sharp transition zones between fresh water and salt water. SUTRA energy transport simulation may be employed to model thermal regimes in aquifers, subsurface heat conduction, aquifer thermal energy storage systems, geothermal reservoirs, thermal pollution of aquifers, and natural hydrogeologic convection systems.

Mesh construction is quite flexible for arbitrary geometries employing quadrilateral finite elements in Cartesian or radial-cylindrical coordinate systems. The mesh may be coarsened employing 'pinch nodes' in areas where transport is unimportant. Permeabilities may be anisotropic and may vary both in direction and magnitude throughout the system as may most other aquifer and fluid properties. Boundary conditions, sources and sinks may be time-dependent. A number of input data checks are made in order to verify the input data set. An option is available for storing intermediate results and restarting simulation at the intermediate time. An option to plot results produces output which may be contoured directly on the printer paper. Options are also available to print fluid velocities in the system, to print fluid mass and solute mass or energy budgets for the system, and to make temporal observations at points in the system.

Both the mathematical basis for SUTRA and the program structure are highly general, and are modularized to allow for straightforward addition of new methods or processes to the simulation. The FORTRAN-77 coding stresses clarity and modularity rather than efficiency, providing easy access for eventual modifications.

ACKNOWLEDGMENTS

The SUTRA computer code and this report were prepared under a joint research project of the U.S. Geological Survey, Department of the Interior (USGS-MIPR-N-83-18) and the Engineering and Services Laboratory, U.S. Air Force Engineering and Services Center (AFESC-JON:2103-9025) entitled, "Groundwater model development for enhanced characterization of contaminant fate and transport."

(p. xi follows)

TABLE OF CONTENTS

SUTRA FUNDAMENTALS

DETAILS OF SUTRA METHODOLOGY

SUTRA SIMULATION EXAMPLES

SUTRA SIMULATION SETUP

LIST OF FIGURES

INTRODUCTION

(p. 3 follows)

Chapter 1

Introduction

1.1 Purpose and Scope

SUTRA (Saturated-Unsaturated Transport) is a computer program which simulates fluid movement and transport of either energy or dissolved substances in a subsurface environment. The model employs a two-dimensional hybrid finite-element and integrated-finite-difference method to approximate the governing equations that describe the two interdependent processes that are simulated:

1) fluid density-dependent saturated or unsaturated ground-water flow, and either

2a) transport of a solute in the ground water, in which the solute may be subject to: equilibrium adsorption on the porous matrix, and both first-order and zero-order production or decay,

or,

2b) transport of thermal energy in the ground water and solid matrix of the aquifer.

SUTRA provides, as the primary calculated result, fluid pressures and either solute concentrations or temperatures, as they vary with time, everywhere in the simulated subsurface system. SUTRA may also be used to simulate simpler subsets of the above processes.

This report describes the physical-mathematical basis and the numerical methodology of the SUTRA computer code. The report may be divided into three levels which may be read depending on the reader's interest. The overview of simulation with SUTRA and methods may be obtained from Chapter 1 - Introduction. The basis, at a fundamental level, for a reader who will carry out simulations with SUTRA may be obtained by additional reading of: Chapter 2 - Physical-Mathematical Basis of SUTRA Simulation, which gives a complete and detailed description of processes which SUTRA simulates and also describes each physical parameter required by SUTRA input data, Chapter 3 - Fundamentals of Numerical Algorithms, which gives an introduction to the numerical aspects of simulation with SUTRA, Chapter 6 - Simulation Examples, and Chapter 7 - Simulation Setup which includes the SUTRA Input Data List. Finally, for complete details of SUTRA methodology, the following additional sections may be read: Chapter 4 - Numerical Methods, and Chapter 5 - Other Methods and Algorithms. Chapter 4 provides the detail upon which program modifications may be based, while portions of Chapter 5 are valuable background for certain simulation applications.

1.2 The Model

SUTRA is based on a general physical, mathematical and numerical structure implemented in the computer code in a modular design. This allows straightforward modifications and additions to the code. Eventual modifications may be, for example, the addition of non-equilibrium sorption (such as two-site models), equilibrium chemical reactions or chemical kinetics, or addition of over- and underburden heat loss functions, a well-bore model, or confining bed leakage.

The SUTRA model stresses general applicability, numerical robustness and accuracy, and clarity in coding. Computational efficiency is somewhat diminished to preserve these qualities. The modular structure of SUTRA, however allows implementation of any eventual changes which may improve efficiency. Such modifications may be in the configuration of the matrix equations, in the solution procedure for these equations, or in the finite-element integration routines. Furthermore, the general nature and flexibility of the input data allows easy adaptability to user-friendly and graphic input-output programming. The modular structure would also ease major changes such as modifications for multi-layer (quasi-three-dimensional) simulations, or for simultaneous energy and solute transport simulations.

SUTRA is primarily intended for two-dimensional simulation of flow, and either solute or energy transport in saturated variable-density systems. While unsaturated flow and transport processes are included to allow simulation of some unsaturated problems, SUTRA numerical algorithms are not specialized for the non-linearities of unsaturated flow as would be required of a model simulating only unsaturated flow. Lacking these special methods, SUTRA requires fine spatial and temporal discretization for unsaturated flow, and is therefore not an economical tool for extensive unsaturated flow modeling. The general unsaturated capability is implemented in SUTRA because it fits simply in the structure of other non-linear coefficients involved in density-dependent flow and transport simulation without requiring special algorithms. The unsaturated flow capability is thus provided as a convenience to the user for occasional analyses rather than as the primary application of this tool.

1.3 SUTRA Processes

Simulation using SUTRA is in two space dimensions, although a three-dimensional quality is provided in that the thickness of the two-dimensional region in the third direction may vary from point to point. Simulation may be done in either the areal plane or in a cross-sectional view. The spatial coordinate system may be either Cartesian (x,y) or radial-cylindrical (r,z). Areal simulation is usually physically unrealistic for variable-density fluid problems.

Ground-water flow is simulated through numerical solution of a fluid mass balance equation. The ground-water system may be either saturated, or partly or completely unsaturated. Fluid density may be constant, or vary as a function of solute concentrations or fluid temperature.

SUTRA tracks the transport of either solute mass or energy in the flowing ground water through a unified equation which represents the transport of either solute or energy. Solute transport is simulated through numerical solution of a solute mass balance equation where solute concentration may affect fluid density. The single solute species may be transported conservatively, or it may undergo equilibrium sorption (through linear, Freundlich or Langmuir isotherms). In addition, the solute may be produced or decay through first- or zero-order processes.

Energy transport is simulated though numerical solution of an energy balance equation. The solid grains of the aquifer matrix and fluid are locally assumed to have equal temperature, and fluid density and viscosity may be affected by the temperature.

Almost all aquifer material, flow, and transport parameters may vary in value throughout the simulated region. Sources and boundary conditions of fluid, solute and energy may be specified to vary with time or may be constant.

SUTRA dispersion processes include diffusion and two types of fluid velocity-dependent dispersion. The standard dispersion model for isotropic media assumes direction-independent values of longitudinal and transverse dispersivity. A velocity-dependent dispersion process for anisotropic media is also provided and is introduced in the SUTRA documentation. This process assumes that longitudinal dispersivity varies depending on the angle between the flow direction and the principal axis of aquifer permeability when permeability is anisotropic.

1.4 Some SUTRA Applications

SUTRA may be employed in one- or two-dimensional analyses. Flow and transport simulation may be either steady-state which requires only a single solution step, or transient which requires a series of time steps in the numerical solution. Single-step steady-state solutions are usually not appropriate for non-linear problems with variable density, saturation, viscosity and non-linear sorption.

SUTRA flow simulation may be employed for areal and cross-sectional modeling of saturated ground-water flow systems, and unsaturated zone flow. Some aquifer tests may be analyzed with flow simulation. SUTRA solute transport simulation may be employed to model natural or man-induced chemical species transport including processes of solute sorption, production and decay. Such simulation may be used to analyze ground-water contaminant transport problems and aquifer restoration designs. SUTRA solute transport simulation may

also be used for modeling of variable density leachate movement, and for cross-sectional modeling of salt-water intrusion in aquifers at both near-well or regional scales with either dispersed or relatively sharp transition zones between fresh water and salt water. SUTRA energy transport simulation may be employed to model thermal regimes in aquifers, subsurface heat conduction, aquifer thermal energy storage systems, geothermal reservoirs, thermal pollution of aquifers, and natural hydrogeologic convection systems.

1.5 SUTRA Numerical Methods

SUTRA simulation is based on a hybridization of finite-element and integrated-finite-difference methods employed in the framework of a method of weighted residuals. The method is robust and accurate when employed with proper spatial and temporal discretization. Standard finite-element approximations are employed only for terms in the balance equations which describe fluxes of fluid mass, solute mass and energy. All other non-flux terms are approximated with a finite-element mesh version of the integrated-finite-difference methods. The hybrid method is the simplest and most economical approach which preserves the mathematical elegance and geometric flexibility of finite-element simulation, while taking advantage of finite-difference efficiency.

SUTRA employs a new method for calculation of fluid velocities. Fluid velocities, when calculated with standard finite-element methods for systems with variable fluid density, may display spurious numerically generated components within each element. These errors are due to fundamental numerical inconsistencies in spatial and temporal approximations for the pressure gradient

and density-gravity terms which are involved in velocity calculation. Spurious velocities can significantly add to the dispersion of solute or energy. This false dispersion makes accurate simulation of all but systems with very low vertical concentration or temperature gradients impossible, even with fine vertical spatial discretization. Velocities as calculated in SUTRA, however, are based on a new, consistent, spatial and temporal discretization, as introduced in this report. The consistently-evaluated velocities allow stable and accurate transport simulation (even at steady state) for systems with large vertical gradients of concentration or temperature. An example of such a system that SUTRA successfully simulates is a cross-sectional regional model of a coastal aquifer wherein the transition zone between horizontally flowing fresh water and deep stagnant salt water is relatively narrow.

The time discretization used in SUTRA is based on a backwards finite-difference approximation for the time derivatives in the balance equations. Some non-linear coefficients are evaluated at the new time level of solution by projection, while others are evaluated at the previous time level for non-iterative solutions. All coefficients are evaluated at the new time level for iterative solutions.

The finite-element method allows the simulation of irregular regions with irregular internal discretization. This is made possible through use of quadrilateral elements with four corner nodes. Coefficients and properties of the system may vary in value throughout the mesh. Manual construction and data preparation for an irregular mesh requires considerable labor, and it may be worthwhile for the user to develop or obtain interactive software for this purpose in the event that irregular mesh construction is often required.

'Pinch nodes' may be introduced in the finite-element mesh to allow for quick changes in mesh size from a fine mesh in the region where transport is of primary interest, to the external region, where only a coarse mesh is needed to define the flow system. Pinch nodes, although simplifying mesh design and reducing the number of nodes required in a particular mesh, also increases the matrix equation band width. Because SUTRA employs a band solver, the increased band width due to the use of pinch nodes may offset the gain in computational efficiency due to fewer nodes. Substitution of a non-band-width-dependent solver would guarantee the advantage that pinch nodes can provide. However, mesh designs employing pinch nodes may be experimented with, using the present solver.

SUTRA includes an optional numerical method based on asymmetric finite element weighting functions which results in 'upstream weighting' of advective transport and unsaturated fluid flux terms. Although upstream weighting has typically been employed to achieve stable, non-oscillatory solutions to transport problems and unsaturated flow problems, the method is not recommended for general use as it merely changes the physical system being simulated by increasing the magnitude of the dispersion process. A practical use of the method is, however, to provide a simulation of the sharpest concentration or temperature variations possible with a given mesh. This is obtained by specifying a simulation with absolutely no physical diffusion or dispersion, and with 50% upstream weighting. The results may be interpreted as the solution with the minimum amount of dispersion possible for a stable result in the particular mesh in use.

In general simulation analyses of transport, upstream weighting is discouraged. The non-upstream methods are also provided by SUTRA, and are based

on symmetric weighting functions. These methods are robust and accurate when the finite-element mesh is properly designed for a particular simulation, and are those which should be used for most transport simulations.

1.6 SUTRA as a Tool of Analysis

SUTRA will provide clear, accurate answers only to well-posed, well-defined, and well-discretized simulation problems. In less-well-defined systems, SUTRA simulation can help visualize a conceptual model of the flow and transport regime, and can aid in deciding between various conceptual models. In such less-well-defined systems, simulation can help answer questions such as: Is the (inaccessible) aquifer boundary which is (probably) ten kilometers offshore either leaky or impermeable? How leaky? Does this boundary affect the primary analysis of onshore water supply?

SUTRA is not useful for making exact predictions of future responses of the typical hydrologic systems which are not well defined. Rather, SUTRA is useful for hypothesis testing and for helping to understand the physics of such a system. On the other hand, developing an understanding of a system based on simulation analysis can help make a set of worthwhile predictions which are predicated on uncertainty of both the physical model design and model parameter values. In particular, transport simulation which relies on large amounts of dispersion must be considered an uncertain basis for prediction, because of the highly idealized description inherent in the SUTRA dispersion process.

A simulation-based prediction made with certainty is often inappropriate, and an "if-then" prediction is more realistic. A reasonable type of result of SUTRA simulation analysis may thus be: "Based on the uncertainty in location

and type of boundary condition A, and uncertainty in the distribution of values for parameters B and C, the following predictions are made. The extreme, but reasonable combination of A, B, and C results in prediction X; the opposite reasonable extreme combination of A, B, and C results in prediction Y; the combination of best estimates of A, B, and C, results in prediction Z, and is considered most likely."

In some cases, the available real data on a system may be so poor that a simulation using SUTRA is so ambiguously defined that no prediction at all can be made. In this instance, the simulation may be used to point out the need for particular types of data collection. The model could be used to advantage in visualizing possible regimes of system behavior rather than to determine which is accurate.

SUTRA FUNDAMENTALS

(p. 15 follows)

Chapter 2

Physical-Mathematical Basis of SUTRA Simulation

The physical mechanisms which drive thermal energy transport and solute transport in the subsurface environment are described by nearly identical mathematical expressions. SUTRA takes advantage of this similarity, and with a simple program structure provides for simulation of either energy or solute transport. In fact, SUTRA simulation combines two physical models, one to simulate the flow of ground water, and the second to simulate the movement of either thermal energy or a single solute in the ground water.

The primary variable upon which the flow model is based is fluid pressure, $p[M/(L \cdot s^2)] = p(x,y,t)$. Pressure may vary spatially in the ground-water system, as well as with time. Pressure is expressed as a combination of fluid mass units, [M], length units ,[L], and time units in seconds, [s]. Fluid density may vary depending on the local value of fluid temperature or solute concentration. Variation in fluid density, aside from fluid pressure differences, may itself drive flows. The effects of gravity acting on fluids with different density must therefore be accounted for in the flow field.

The flow of ground water, in turn, is a fundamental mechanism upon which the physical models of energy transport and solute transport are based. The primary variable characterizing the thermal energy distribution in a ground-water system is fluid temperature, $T[°C] = T(x,y,t)$, in degrees Celcius, which may vary spatially and with time. The primary variable characterizing the state of solute distribution in a ground-water system is solute mass fraction, $C[M_s/M] = C(x,y,t)$, which may also vary spatially and with time. The units are a ratio of solute mass, $[M_s]$ to fluid mass, [M]. The term 'solute mass fraction'

may be used interchangeably with 'solute concentration', and no difference should be implied. Note that 'solute volumetric concentration', $c[M_s/L_f^3]$, (mass of solute, M_s, per volume of fluid, L_f^3), is not the primary variable characterizing solute transport referred to either in this report or in output from the SUTRA model. Note that the measure of solute mass $[M_s]$ may be in units such as [mg], [kg], [moles], or [lbm], and may differ from the measure, [M], of fluid mass.

SUTRA allows only the transport of either thermal energy or a single solute to be modeled in a given simulation. Thus, when simulating energy transport, a constant value of solute concentration is assumed in the ground water. When simulating solute transport, a constant ground-water temperature is assumed.

SUTRA simulation is carried out in two space dimensions with parameters varying in these two directions. However, the region of space to be simulated may be defined as three dimensional, when the assumption is made that all SUTRA parameters and coefficients have a constant value in the third space direction. A SUTRA simulation may be carried out over a region defined over two space coordinates (x,y) in which the thickness of the region measured in the third coordinate direction (z) varies depending on (x,y) position.

2.1 Physical Properties of Solid Matrix and Fluid

Fluid physical properties

The ground-water fluid density and viscosity may vary depending on pressure, temperature and solute concentration. These fundamental variables are defined as follows:

$p(x,y,t)$	$\lvert M/(L \cdot s^2) \rvert$	fluid pressure
$T(x,y,t)$	$\lvert °C \rvert$	fluid temperature (degrees Celcius)
$C(x,y,t)$	$\lvert M_s/M \rvert$	fluid solute mass fraction (or solute concentration) (mass solute per mass total fluid)

As a point of reference, the 'solute volumetric concentration' is defined in terms of fluid density, ρ:

$c(x,y,t)$	$\lvert M_s/L_f^3 \rvert$	solute volumetric concentration (mass solute per volume total fluid)
$\rho(x,y,t)$	$\lvert M/L_f^3 \rvert$	fluid density

$$c = \rho C \tag{2.1}$$

$$\rho = \rho_w + c \tag{2.2}$$

Total fluid density is the sum of pure water density, ρ_w, and c. Note again that 'solute concentration' refers to solute mass fraction, C, and not c. Fluid density, while a weak function of pressure is primarily dependent upon fluid solute concentration and temperature. The approximate density models employed by SUTRA are first order Taylor expansions about a base (reference) density other density models may be substituted through minor modifications to the program. For energy transport:

$$\rho = \rho(T) \simeq \rho_o + \frac{\partial \rho}{\partial T} (T - T_o) \tag{2.3}$$

ρ_o	$\lvert M/L_f^3 \rvert$	base fluid density at $T=T_o$
T_o	$\lvert °C \rvert$	base fluid temperature

where ρ_o is the base fluid density at a base (reference) temperature of T_o, and $\partial\rho/\partial T$ is a constant value of density change with temperature. For the

range 20°C to 60°C, $\partial\rho/\partial T$ is approximately -.375 [kg/(m^3·°C)]; however, this factor varies and should be carefully chosen for the temperature range of interest.

For solute transport:

$$\rho = \rho(C) \simeq \rho_o + \frac{\partial \rho}{\partial C} (C - C_o) \qquad (2.4)$$

ρ_o	$[M/L_f^3]$	base fluid density at $C=C_o$
C_o	$[M_s/M]$	base fluid solute concentration

where ρ_o is the base fluid density at base concentration, C_o. (Usually, $C_o = 0$, and the base density is that of pure water.) The factor $\partial\rho/\partial C$ is a constant value of density change with concentration. For example, for mixtures of fresh and sea water at 20°C, when C is the mass fraction of total dissolved solids, $C_o = 0$, and $\rho_o = 998.2$ [kg/m^3], then the factor, $\partial\rho/\partial C$, is approximately 700. [kg/m^3].

Fluid viscosity, μ [$M/L_f \cdot s$], is a weak function of pressure and of concentration, (for all except very high concentrations), and depends primarily on fluid temperature. For energy transport the viscosity of pure water is given in m-k-s units by:

$$\mu(T) \simeq (239.4 \times 10^{-7})\ 10^{\left(\frac{248.37}{T+133.15}\right)} \ [kg/(m \cdot s)] \qquad (2.5)$$

(The units may be converted to those desired via a scale factor in the program input data.)

For solute transport, viscosity is taken to be constant. For example, at 20°C in m-k-s units:

$$\mu(C)\Big|_{T = 20°C} = 1.0 \times 10^{-3} \ [kg/(m \cdot s)] \qquad (2.6)$$

Properties of fluid within the solid matrix

The total volume of a porous medium is composed of a matrix of solid grains typically of solid earth materials, and of void space which includes the entire remaining volume which the solid does not fill. The volume of void space may be fully or partly filled with gas or liquid, and is commonly referred to as the pore volume. Porosity is defined as a volume of voids in the soil matrix per total volume of voids plus matrix:

$\varepsilon(x,y,t)$	[1]	porosity (volume of voids per total volume)

where [1] indicates a dimensionless quantity.

It should be noted that SUTRA employs only one type of porosity, ε. In some instances there may be need to distinguish between a porosity for pores which take part in fluid flow, and pores which contain stagnant fluid. (Modifications may be made by the user to include this process.)

The fraction of total volume filled by the fluid is εS_w where:

$S_w(x,y,t)$	[1]	water saturation (saturation) (volume of water per volume of voids)

When $S_w = 1$, the void space is completely filled with fluid and is said to be saturated. When $S_w < 1$, the void space is only partly water filled and is referred to as being unsaturated.

When $S_w < 1$, water adheres to the surface of solid grains by surface tension effects, and the fluid pressure is less than atmospheric. Fluid pressure, p, is measured with respect to background or atmospheric pressure. The negative pressure is defined as capillary pressure, which exists only for $p < 0$:

$p_c(x,y,t)$ $[M/(L \cdot s^2)]$ capillary pressure

$p_c = -p$ when $p < 0$

$p_c = 0$ when $p \geq 0$ (2.7)

In a saturated porous medium, as fluid (gauge) pressure drops below zero, air may not directly enter the void space, but may enter suddenly when a critical capillary pressure is reached. This pressure, p_{cent}, is the entry pressure (or bubble pressure):

p_{cent} $[M/(L \cdot s^2)]$ entry capillary pressure

Typical values for p_{cent} range from about 1. x 10^3 $[kg/(m \cdot s^2)]$ for coarse sand to approximately 5. x 10^3 $[kg/(m \cdot s^2)]$ for fine silty sand.

The relationship between fluid saturation and capillary pressure in a given medium is typically determined by laboratory experiment, and except for the portion near bubble pressure, tends to have an exponential character (Figure 2.1). Different functional relationships exists for different materials as measured in the laboratory. Also a number of general functions with parameters to be fitted to laboratory data are available. Because of the variety of possible functions, no particular function is set by SUTRA; any desired function may be specified for simulation of unsaturated flow. For example, a general function with three fitting parameters is (Van Genuchten, 1980):

$$S_w = S_{wres} + \left(1-S_{wres}\right)\left[\frac{1}{1 + (ap_c)^n}\right]^{\left(\frac{n-1}{n}\right)} \qquad (2.8)$$

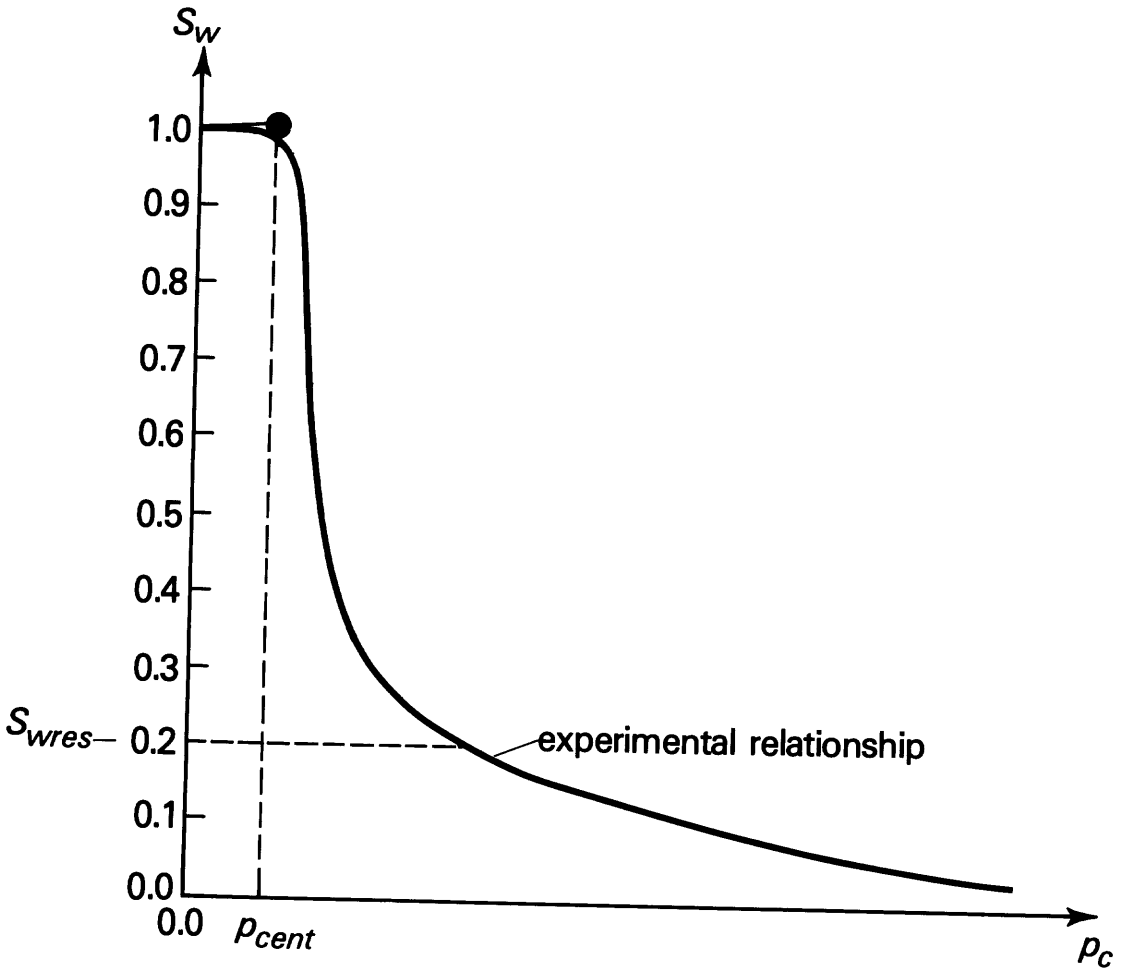


Figure 2.1
Saturation-capillary pressure relationship (schematic).

where S_{wres} is a residual saturation below which saturation is not expected to fall (because the fluid becomes immobile), and both a and n are parameters. The values of these parameters depend upon a number of factors and must be carefully chosen for a particular material.

The total mass of fluid contained in a total volume, VOL, of solid matrix plus pore space is $(\varepsilon S_w \rho)$VOL. The actual amount of total fluid mass contained depends solely on fluid pressure, p, and solute concentration, C, or fluid temperature, T. A change in total fluid mass in a volume, assuming VOL is constant, is expressed as follows:

$$\text{VOL}\cdot d(\varepsilon S_w \rho) = \text{VOL} \cdot \left[\frac{\partial(\varepsilon S_w \rho)}{\partial p}\, dp + \frac{\partial(\varepsilon S_w \rho)}{\partial U}\, dU\right] \qquad (2.9)$$

where U represents either C or T. Saturation, S_w, is entirely dependent on fluid pressure, and porosity, ε, does not depend on concentration or temperature:

$$\text{VOL}\cdot d(\varepsilon S_w \rho) = \text{VOL}\cdot\left[\left(S_w \frac{\partial(\varepsilon\rho)}{\partial p} + \varepsilon\rho \frac{\partial S_w}{\partial p}\right)dp + \varepsilon S_w \frac{\partial \rho}{\partial U}\, dU\right] \qquad (2.10)$$

The factor, $\partial S_w/\partial p$, is obtained by differentiation of the chosen saturation-capillary pressure relationship. For the example function given as (2.8),

$$\frac{dS_w}{dp} = \frac{a(n-1)\left(1-S_{wres}\right)\left(ap_c\right)^{(n-1)}}{\left(1 + (ap_c)^n\right)\left(\frac{2n-1}{n}\right)} \qquad (2.11)$$

The factor, $\partial\rho/\partial U$, is a constant value defined by the assumed density models, given by equations (2.3) and (2.4).

Aquifer storativity under fully saturated conditions is related to the factor, $\partial(\varepsilon\rho)/\partial p$, by definition, as follows (Bear, 1979):

$$\frac{\partial(\varepsilon\rho)}{\partial p} \equiv \rho S_{op} \qquad (2.12)$$

where:

$$S_{op} \equiv \frac{1}{VOL}\left(\frac{\Delta VOL_w}{\Delta p}\right) \qquad (2.13)$$

$S_{op}(x,y)$ $\quad [M/(L\cdot s^2)]^{-1}$ $\quad$ specific pressure storativity

The specific pressure storativity, S_{op}, is the volume of water released from saturated pore storage due to a unit drop in fluid pressure per total solid matrix plus pore volume. Note that the common specific storativity, S_o $[L^{-1}]$, which when multiplied by confined aquifer thickness gives the well known storage coefficient, S[1], is related to S_{op} as, $S_o = \rho|g|S_{op}$, where $|g|[L/s^2]$ is the magnitude of the gravitational acceleration vector. The common specific storativity, S_o, is analagous to specific pressure storativity, S_{op}, used in SUTRA, except that S_o expresses the volume of water released from pore storage due to a unit drop in piezometric head.

SUTRA employs an expanded form of the specific pressure storativity based on fluid and bulk porous matrix compressibilities. The relationship is obtained as follows by expanding equation (2.12)

$$\rho\, S_{op} = \rho \frac{\partial\varepsilon}{\partial p} + \varepsilon \frac{\partial\rho}{\partial p} \qquad (2.14)$$

The coefficient of compressibility of water is defined by

$$\beta \equiv \frac{1}{\rho}\frac{\partial\rho}{\partial p} \qquad (2.15)$$

β $\quad [M/(L\cdot s^2)]^{-1}$ $\quad$ fluid compressibility

which allows the last term of (2.14) to be replaced by $\varepsilon\rho\beta$. For pure water at 20°C, $\beta \sim 4.47 \times 10^{-10}$ $[kg/(m \cdot s^2)]^{-1}$. As the volume of solid grains VOL_s, in a volume, VOL, of porous solid matrix plus void space is $VOL_s = (1-\varepsilon) \cdot VOL$, the factor, $\partial\varepsilon/\partial p$, may be expressed as:

$$\frac{\partial \varepsilon}{\partial p} = \frac{(1-\varepsilon)}{VOL} \frac{\partial (VOL)}{\partial p} \tag{2.16}$$

which assumes that individual solid grains are relatively incompressible. The total stress at any point in the solid matrix-fluid system is the sum of effective (intergranular) stress, σ' $[M/(L \cdot s^2)]$, and fluid pore pressure, p. In systems where the total stress remains nearly constant, $d\sigma' = -dp$, and any drop in fluid pressure increases intergranular stress by a like amount. This consideration allows (2.16) to be expressed in terms of bulk porous matrix compressibility, as: $\partial\varepsilon/\partial p = (1-\varepsilon)\alpha$, where

$$\alpha \equiv - \frac{1}{VOL} \frac{\partial (VOL)}{\partial \sigma'} \tag{2.17}$$

α	$[M/(L \cdot s^2)]^{-1}$	porous matrix compressibility
σ'	$[M/(L \cdot s^2)]$	intergranular stress

Factor α ranges from $\alpha \sim 10^{-10}$ $[kg/(m \cdot s^2)]^{-1}$ for sound bedrock to about $\alpha \sim 10^{-7}$ $[kg/(m \cdot s^2)]^{-1}$ for clay. Thus equation (2.14) may be rewritten as $\rho S_{op} = \rho(1-\varepsilon)\alpha + \varepsilon\rho\beta$, and, in effect, the specific pressure storativity, S_{op}, is expanded as:

$$S_{op} = (1-\varepsilon)\alpha + \varepsilon\beta \tag{2.18}$$

A more thorough discussion of storativity is presented by Bear (1979).

2.2 Description of Saturated-Unsaturated Ground-water Flow

Fluid flow and flow properties

Fluid movement in porous media where fluid density varies spatially may be driven by either differences in fluid pressure or by unstable variations in fluid density. Pressure-driven flows, for example, are directed from regions of higher than hydrostatic fluid pressure toward regions of lower than hydrostatic pressure. Density-driven flows occur when gravity forces act on denser regions of fluid causing them to flow downward relative to fluid regions which are less dense. A stable density configuration drives no flow, and is one in which fluid density remains constant or increases with depth.

The mechanisms of pressure and density driving forces for flow are expressed for SUTRA simulation by a general form of Darcy's law which is commonly used to describe flow in porous media:

$$\underline{v} = -\left(\frac{\underline{\underline{k}}k_r}{\epsilon S_w \mu}\right) \cdot (\underline{\nabla} p - \rho \underline{g}) \qquad (2.19a)$$

where:

$\underline{v}$ (x,y,t)	[L/s]	average fluid velocity
$\underline{\underline{k}}$ (x,y)	[L^2]	solid matrix permeability (2 X 2 tensor of values)
k_r(x,y,t)	[1]	relative permeability to fluid flow (assumed to be independent of direction.)
$\underline{g}$	[L/s^2]	gravitational acceleration (gravity vector) (1 x 2 vector of values)

The gravity vector is defined in relation to the direction in which vertical elevation is measured:

$$\underline{g} = -|\underline{g}|\ \underline{\nabla}(\text{ELEVATION}) \qquad (2.19b)$$

where $|\underline{g}|$ is the magnitude of the gravitational acceleration vector. For example, if the y-space-coordinate is oriented directly upwards, then $\underline{\nabla}$(ELEVATION) is a vector of values (for x and y directions, respectively): (0,1), and $g = (0,-|\underline{g}|)$. If for example, ELEVATION increases in the x-y plane at a 60° angle to the x-axis, then $\underline{\nabla}(\text{ELEVATION}) = ((1/2), (3^{\frac{1}{2}}/2))$ and $\underline{g} = (-(1/2)|\underline{g}|, -(3^{\frac{1}{2}}/2)|\underline{g}|)$.

The average fluid velocity, $\underline{v}$, is the velocity of fluid with respect to the stationary solid matrix. The so-called Darcy velocity, $\underline{q}$, for the sake of reference, is $\underline{q} = \varepsilon S_w \underline{v}$. This value is always less than the true average fluid velocity, $\underline{v}$, and thus, not being a true indicator of the speed of water movement, 'Darcy velocity', $\underline{q}$, is not a useful concept in simulation of subsurface transport. The velocity is referred to as an 'average', because true velocities in a porous medium vary from point to point due to variations in the permeability and porosity of the medium at a spatial scale smaller than that at which measurements were made.

Fluid velocity, even for a given pressure and density distribution, may take on different values depending on how mobile the fluid is within the solid matrix. Fluid mobility depends on the combination of permeability, $\underline{\underline{k}}$, relative permeability, k_r, and viscosity, μ, that occurs in equation (2.19a). Permeability is a measure of the ease of fluid movement through interconnected voids in the solid matrix when all voids are completely saturated. Relative permeability expresses what fraction of the total permeability remains when the voids are only partly fluid-filled and only part of the total interconnected void space is, in fact, connected by continuous fluid channels. Viscosity directly expresses ease of fluid flow; a less viscous fluid flows more readily under a driving force.

As a point of reference, in order to relate the general form of Darcy's law, (2.19a), back to a better-known form dependent on hydraulic head, the dependence of flow on density and saturation must be ignored. When the solid matrix is fully saturated, $S_w = 1$, the relative permeability to flow is unity $k_r = 1$. When, in addition, fluid density is constant, the right side of (2.19a) expanded by (2.19b) may be multiplied and divided by $\rho|\underline{g}|$:

$$\underline{v} = \frac{-\underline{\underline{k}}\rho|\underline{g}|}{\varepsilon\mu} \cdot [\underline{\nabla}\,(\frac{p}{\rho|\underline{g}|}) + \underline{\nabla}\,(\text{ELEVATION})] \qquad (2.20a)$$

The hydraulic conductivity, $\underline{\underline{K}}\,(x,y,t)$ [L/s], may be identified in this equation as, $\underline{\underline{K}}=(\underline{\underline{k}}\rho|\underline{g}|)/\mu$; pressure head, $h_p(x,y,t)$ [L], is $h_p = p/(\rho|\underline{g}|)$. Hydraulic head, $h(x,y,t)$ [L], is $h = h_p + \text{ELEVATION}$. Thus, for constant density, saturated flow:

$$\underline{v} = -\left(\frac{\underline{\underline{K}}}{\varepsilon}\right) \cdot \underline{\nabla}\, h \qquad (2.20b)$$

which is Darcy's law written in terms of the hydraulic head. Even in this basic form of Darcy's law, flow may depend on solute concentration and temperature. The hydraulic conductivity, through viscosity, is highly dependent on temperature, and measurably, but considerably less on concentration. In cases where density or viscosity are not constant, therefore, hydraulic conductivity, $\underline{\underline{K}}$, is not a fundamental parameter describing ease of flow through the solid matrix. Permeability, $\underline{\underline{k}}$, is in most situations, essentially independent of pressure, temperature and concentration and therefore is the appropriate fundamental parameter describing ease of flow in the SUTRA model.

Permeability, $\underline{\underline{k}}$, describes ease of fluid flow in a saturated solid matrix. When permeability to flow in a particular small volume of solid matrix differs depending upon in which direction the flow occurs, the permeability is said to be anisotropic. Direction-independent permeability is called isotropic. It is commonly assumed that permeability is the same for flow forward or backward along a particular line in space. When permeability is anisotropic, there is always one particular direction, x_p, along which permeability has an absolute maximum value, k_{max} $[L^2]$. Somewhere in the plane perpendicular to the 'maximum direction' there is a direction, x_m, in which permeability has the absolute minimum value, $k_{min}[L^2]$, which exists for the particular volume of solid matrix. Thus, in two dimensions, there are two principal orthogonal directions of anisotropic permeability. Both principal directions, x_p and x_m, are assumed to be within the (x,y) plane of the two-dimensional model.

The permeability tensor, $\underline{\underline{k}}$, of Darcy's law, equation (2.19) has four components in two dimensions. These tensorial components have values which depend on effective permeabilities in the x and y coordinate directions which are not necessarily exactly aligned with the principal directions of permeability. The fact that maximum and minimum principal permeability values may change in both value and direction from place to place in the modeled region makes the calculation of the permeability tensor, which is aligned in x and y, complex. The required coordinate rotations are carried out automatically by SUTRA according to the method described in section 5.1, "Rotation of Permeability Tensor".

An anisotropic permeability field in two dimensions is completely described by the values k_{max} and k_{min}, and the angle orienting the principal directions, x_p and x_m, to the x and y directions through the permeability ellipse shown in Figure 2.2. The semi-major and semi-minor axes of the ellipse are defined as

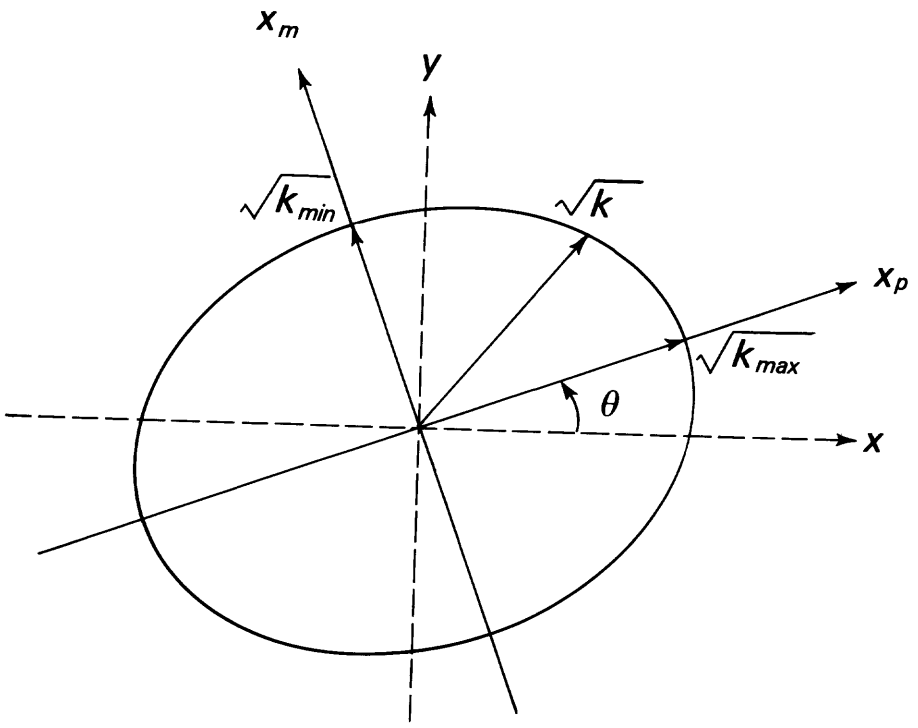


Figure 2.2
Definition of anisotropic permeability and effective permeability, k.

$k_{max}^{\frac{1}{2}}$, and $k_{min}^{\frac{1}{2}}$, respectively, and the length of any radius is $k^{\frac{1}{2}}$, where k is the effective permeability for flow along that direction. Only, k_{max}, k_{min}, and θ, the angle between the x-axis and the maximum direction x_p need be specified to define the permeability, k, in any direction, where:

$k_{max}(x,y)$	$[L^2]$	absolute maximum value of permeability
$k_{min}(x,y)$	$[L^2]$	absolute minimum value of permeability
$\theta(x,y)$	[°]	angle from +x-coordinate axis to direction of maximum permeability, x_p

In the case of isotropic permeability, $k_{max} = k_{min}$, and θ is arbitrary.

The discussion of isotropic and anisotropic permeability, $\underline{\underline{k}}$, applies as well to flow in an unsaturated solid matrix, $S_w < 1$, although unsaturated flow has additional unique properties which require definition. When fluid capillary pressure, p_c, is less than entry capillary pressure, p_{cent}, the void space is saturated $S_w = 1$, and local porous medium flow properties are not pressure-dependent but depend only on void space geometry and connectivity. When $p_c > p_{cent}$, then air or another gas has entered the matrix and the void space is only partly fluid filled, $S_w < 1$. In this case, the ease with which fluid can pass through the solid matrix depends on the remaining cross-section of well-connected fluid channels through the matrix, as well as on surface tension forces at fluid-gas, and fluid-solid interfaces. When saturation is so small such that no interconnected fluid channels exist and residual fluid is scattered about and tightly bound in the smallest void spaces by surface tension, flow ceases entirely. The relative permeability to flow, k_r, which is a measure of this behavior, varies from a value of zero or near-zero at the residual fluid

saturation, S_{wres}, to a value of one at saturation, $S_w = 1$. A relative permeability-saturation relationship (Figure 2.3) is typically determined for a particular solid matrix material in the laboratory as is the relationship, $S_w(p_c)$. Relative permeability is assumed in SUTRA to be independent of direction in the porous media.

SUTRA allows any desired function to be specified which gives the relative permeability in terms of saturation or pressure. A general function, for example, based on the saturation-capillary pressure relationship given as an example in (2.8) is (Van Genuchten, 1980):

$$k_r = S_w^{*\frac{1}{2}} \left\{ 1 - \left[1 - S_w^{*\left(\frac{n}{n-1}\right)} \right]^{\left(\frac{n-1}{n}\right)} \right\}^2 \qquad (2.21a)$$

where the a dimensionless saturation, S_w^*, is given by:

$$S_w^* = \frac{S_w - S_{wres}}{1 - S_{wres}} \qquad (2.21b)$$

Flow in the gaseous phase that fills the remaining void space not containing fluid when $S_w < 1$ is assumed not to contribute significantly to total solute or energy transport which is due primarily to fluid flow and other transport processes through both fluid and solid matrix. Furthermore it is assumed that pressure differences within the gas do not drive significant fluid flow. These assumptions are justified in most common situations when gas pressure is approximately constant throughout the solid matrix system. Should gas pressure vary appreciably in a field system, simulation with SUTRA, which is by definition a single phase flow and transport model, must be critically evaluated against the possible necessity of employing a multiphase fluid flow and transport model.

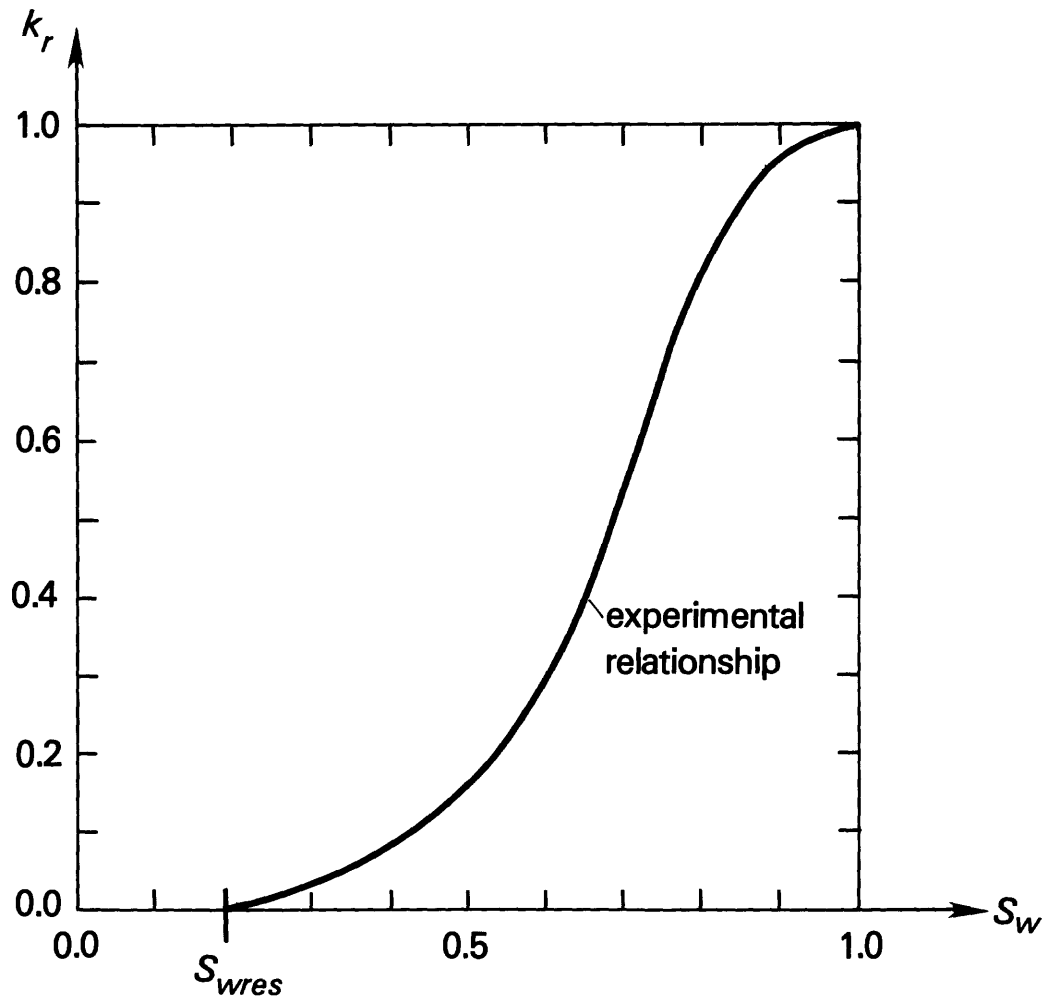


Figure 2.3
Relative permeability-saturation relationship (schematic).

Fluid mass balance

The "so-called" flow simulation provided by SUTRA is in actuality a calculation of how the amount of fluid mass contained within the void spaces of the solid matrix changes with time. In a particular volume of solid matrix and void space, the total fluid mass $(\varepsilon S_w \rho) \cdot VOL$, may change with time due to: ambient ground-water inflows or outflows, injection or withdrawal wells, changes in fluid density caused by changing temperature or concentration, or changes in saturation. SUTRA flow simulation is, in fact, a fluid mass balance which keeps track of the fluid mass contained at every point in the simulated ground-water system as it changes with time due to flows, wells, and saturation or density changes.

The fluid mass balance is expressed as the sum of pure water and pure solute mass balances for a solid matrix in which there is negligible net movement:

$$\frac{\partial(\varepsilon S_w \rho)}{\partial t} = - \underline{\nabla} \cdot (\varepsilon S_w \rho \underline{v}) + Q_p + T \qquad (2.22)$$

where:

$Q_p(x,y,t)$	$[M/(L^3 \cdot s)]$	fluid mass source (including pure water mass plus solute mass dissolved in source water)
$T(x,y,t)$	$[M/(L^3 \cdot s)]$	solute mass source (e.g., dissolution of solid matrix or desorption)

The term on the left may be recognized as the total change in fluid mass contained in the void space with time. The term involving $\underline{\nabla}$ represents contributions to local fluid mass change due to excess of fluid inflows over outflows at a point. The fluid mass source term, Q_p, accounts for external additions of fluid including pure water mass plus the mass of any solute dissolved in the source fluid. The pure solute mass source term, T, may account for external additions

of pure solute mass not associated with a fluid source. In most cases, this contribution to the total mass is small compared to the total pure water mass contributed by fluid sources, Q_p. Pure solute sources, T, are therefore neglected in the fluid mass balance, but may be readily included in SUTRA for special situations. Note that solute mass sources are not neglected in the solute mass balance, which is discussed in section 2.4.

While (2.22) is the most fundamental form of the fluid mass balance, it is necessary to express each mechanism represented by a term of the equation, in terms of the primary variables, p, C, and T. As SUTRA allows variation in only one of C or T at a time, the letter U is employed to represent either of these quantities. The development from equation (2.9) to (2.18) allows the time derivative in (2.22) to be expanded:

$$\frac{\partial(\varepsilon S_w \rho)}{\partial t} = (S_w \rho S_{op} + \varepsilon\rho\frac{\partial S_w}{\partial p})\frac{\partial p}{\partial t} + (\varepsilon S_w \frac{\partial \rho}{\partial U})\frac{\partial U}{\partial t} \qquad (2.23)$$

While the concepts upon which specific pressure storativity, S_{op}, is based, do not exactly hold for unsaturated media, the error introduced by summing the storativity term with the term involving $(\partial S_w/\partial p)$ is insignificant as $(\partial S_w/\partial p) >>> S_{op}$.

The exact form of the fluid mass balance as implemented in SUTRA is obtained from (2.22) by neglecting T, substituting (2.23) and employing Darcy's law, (2.19), for $\underline{v}$:

$$\left(S_w \rho S_{op} + \varepsilon\rho\frac{\partial S_w}{\partial p}\right)\frac{\partial p}{\partial t} + \left(\varepsilon S_w \frac{\partial \rho}{\partial U}\right)\frac{\partial U}{\partial t} - \underline{\nabla}\cdot\left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right)\cdot\left(\underline{\nabla}p - \rho\underline{g}\right)\right] = Q_p \qquad (2.24)$$

2.3 Description of Energy Transport in Ground Water

Subsurface energy transport mechanisms

Energy is transported in the water-solid matrix system by flow of ground water, and by thermal conduction from higher to lower temperatures through both the fluid and solid. The actual flow velocities of the ground water from point to point in the three-dimensional space of an aquifer may vary considerably about an average two-dimensional velocity uniform in the z-direction, $\underline{v}(x,y,t)$, calculated from Darcy's law (2.22). As the true, not-average, velocity field is usually too complex to measure in real systems, an additional transport mechanism approximating the effects of mixing of different temperature ground waters moving both faster and slower than average velocity, $\underline{v}$, is hypothesized. This mechanism, called energy dispersion, is employed in SUTRA as the best currently available, though approximate description, of the mixing process. In the simple dispersion model employed, dispersion, in effect, adds to the thermal conductivity value of the fluid-solid medium in particular directions dependent upon the direction of fluid flow. In other words, mixing due to the existence of non-uniform, nonaverage velocities in three dimensions about the average-uniform flow, $\underline{v}$, is conceptualized in two dimensions as a diffusion-like process with anisotropic diffusivities.

The model has, in fact, been shown to describe transport well in purely homogeneous porous media with uniform one-dimensional flows. In heterogeneous field situations with non-uniform flow in, for example, irregular bedding or fractures, the model holds only at the pre-determined scale at which dispersivities are calibrated and it must be considered as a currently necessary approximation, and be carefully applied when extrapolating to other scales of transport.

Solid matrix-fluid energy balance

The simulation of energy transport provided by SUTRA is actually a calculation of the time rate of change of the amount of energy stored in the solid matrix and fluid. In any particular volume of solid matrix plus fluid, the amount of energy contained is $[\varepsilon S_w \rho e_w + (1-\varepsilon)\rho_s e_s]\cdot VOL$, where

e_w	$[E/M]$	energy per unit mass water
e_s	$[E/M_G]$	energy per unit mass solid matrix
ρ_s	$[M_G/L_G^3]$	density of solid grain in solid matrix

and where [E] are energy units $[M\cdot L^2/s^2]$.

The stored energy in a volume may change with time due to: ambient water with a different temperature flowing in, well water of a different temperature injected, changes in the total mass of water in the block, thermal conduction (energy diffusion) into or out of the volume, and energy dispersion in or out.

This balance of changes in stored energy with various energy fluxes is expressed as follows:

$$\frac{\partial[\varepsilon S_w \rho e_w + (1-\varepsilon)\rho_s e_s]}{\partial t} = -\underline{\nabla}\cdot(\varepsilon S_w \rho e_w \underline{v}) + \underline{\nabla}\cdot[\lambda \underline{\underline{I}}\cdot\underline{\nabla}T]$$

$$+ \underline{\nabla}\cdot[\varepsilon S_w \rho c_w \underline{\underline{D}}\cdot\underline{\nabla}T] + Q_p c_w T^* + \varepsilon S_w \rho \gamma_o^w + (1-\varepsilon)\rho_s \gamma_o^s \qquad (2.25)$$

$\lambda(x,y,t)$	$[E/(s\cdot L\cdot °C)]$	bulk thermal conductivity of solid matrix plus fluid
$\underline{\underline{I}}$	[1]	identity tensor (ones on diagonal, zeroes elsewhere) (2x2)
c_w	$[E/(M\cdot °C)]$	specific heat of water ($c_w \sim 4.182 \times 10^3$ [J/(kg·°C)] at 20°C)

$\underline{\underline{D}}(x,y,t)$	$[L^2/s]$	dispersion tensor (2 X 2)
$T^*(x,y,t)$	[°C]	temperature of source fluid
$\gamma_o^w(x,y,t)$	[E/M·s]	energy source in fluid
$\gamma_o^s(x,y,t)$	$[E/M_G \cdot s]$	energy source in solid grains

The time derivative expresses the total change in energy stored in both the solid matrix and fluid per unit total volume. The term involving $\underline{v}$ expresses contributions to locally stored energy from average-uniform flowing fluid (average energy advection). The term involving bulk thermal conductivity, λ, expresses heat conduction contributions to local stored energy and the term involving the dispersivity tensor, $\underline{\underline{D}}$, approximately expresses the contribution of irregular flows and mixing which are not accounted for by average energy advection. The term involving $_pQ$ accounts for the energy added by a fluid source with temperature, T^*. The last terms account for energy production in the fluid and solid, respectively, due to endothermic reactions, for example.

While more complex models are available and may be implemented if desired, SUTRA employs a volumetric average approximation for bulk thermal conductivity, λ:

$$\lambda \equiv \varepsilon S_w \lambda_w + (1-\varepsilon)\lambda_s \qquad (2.26)$$

λ_w	[E/(s·L·°C)]	fluid thermal conductivity (λ_w ~ 0.6 [J/(s·m·°C)] at 20°C)
λ_s	[E/(s·L·°C)]	solid thermal conductivity (λ_s ~ 3.5 [J/(s·m·°C)] at 20°C for sandstone)

The specific energy content (per unit mass) of the fluid and the solid matrix depends on temperature as follows:

$$e_w = c_w T \tag{2.27a}$$

$$e_s = c_s T \tag{2.27b}$$

c_s	$[E/(M_G \cdot °C)]$	solid grain specific heat ($c_s \sim 8.4 \times 10^2 [J/(kg \cdot °C)]$ for sandstone at 20°C)

An expanded form of the solid matrix-fluid energy balance is obtained by substitution of (2.27a,b) and (2.26) into (2.25). This yields:

$$\begin{aligned} &\frac{\partial}{\partial t}[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s]T + \underline{\nabla}\cdot(\varepsilon S_w \rho c_w \underline{v} T) \\ &- \underline{\nabla}\cdot\{[S_w \lambda_w + (1-\varepsilon)\lambda_s]\,\underline{\underline{I}} + \varepsilon S_w \rho c_w \underline{\underline{D}}\}\cdot\underline{\nabla}T] \\ &= Q_p c_w T^* + \varepsilon S_w \rho \gamma_o^w + (1-\varepsilon)\rho_s \gamma_o^s \end{aligned} \tag{2.28}$$

2.4 Description of Solute Transport in Ground Water

Subsurface solute transport mechanisms

Solute mass is transported through the porous medium by flow of ground water (solute advection) and by molecular or ionic diffusion, which while small on a field scale, carries solute mass from areas of high to low concentrations. The actual flow velocities of the ground water from point to point in three-dimensional space of an aquifer may vary considerably about an average, uniform two-dimensional velocity, $\underline{v}$, which is calculated from Darcy's law (2.22). As the true, not-average, velocity field is usually too complex to measure in real systems, an additional transport mechanism approximating the effects of mixing of waters with different concentrations moving both faster and slower than the average velocity, $\underline{v}(x,y,t)$, is hypothesized. This mechanism, called solute dispersion, is employed in SUTRA as the best currently available, though approximate, description of the mixing process. In the simple dispersion model

employed, dispersion, in effect, significantly adds to the molecular diffusivity value of the fluid in particular directions dependent upoon the direction of fluid flow. In other words, mixing due to the existence of non-uniform, non-average velocities in three dimensions about the average flow, $\underline{v}$, is conceptualized in two dimensions, as a diffusion-like process with anisotropic diffusivities.

The model has, in fact, been shown to describe transport well in purely homogeneous porous media with uniform one-dimensional flows. In heterogeneous field situations with non-uniform flows in, for example, irregular bedding or fractures, the model holds only at the pre-determined scale at which dispersivities are calibrated and it must be considered as a currently necessary approximation, and be carefully applied when extrapolating to other scales of transport.

Solute and adsorbate mass balances

SUTRA solute transport simulation accounts for a single species mass stored in fluid solution as solute and species mass stored as adsorbate on the surfaces of solid matrix grains. Solute concentration, C, and adsorbate concentration, $C_s(x,y,t)$ $[M/M_G]$, (where [M] denotes units of solute mass, and $[M_G]$ denotes units of solid grain mass), are related through equilibrium adsorption isotherms. The species mass stored in solution in a particular volume of solid matrix may change with time due to ambient water with a different concentration flowing in, well water injected with a different concentration, changes in the total fluid mass in the block, solute diffusion or dispersion in or out of the volume, transfer of dissolved species to adsorbed species (or reverse), or a chemical or biological reaction causing solute production or decay. The species mass stored as

adsorbate on the surface of solid grains in a particular block of solid matrix may change with time due to a gain of adsorbed species by transfer of solute from the fluid (or reverse), or a chemical or biological reaction causing adsorbate production or decay.

The separate balances for a single species stored in solution (solute) and on the solid grains (adsorbate), are expressed, respectively, as follows:

$$\frac{\partial(\varepsilon S_w \rho C)}{\partial t} = -f - \underline{\nabla}\cdot(\varepsilon S_w \rho \underline{v} C) + \underline{\nabla}\cdot[\varepsilon S_w \rho(D_m \underline{\underline{I}} + \underline{\underline{D}})\cdot\underline{\nabla}C] + \varepsilon S_w \rho \Gamma_w + Q_p C^* \tag{2.29}$$

$$\frac{\partial[(1-\varepsilon)\rho_s C_s]}{\partial t} = +f + (1-\varepsilon)\rho_s \Gamma_s \tag{2.30}$$

$f(x,y,t)$	$[M_s/(L^3\cdot s)]$	volumetric adsorbate source (gain of absorbed species by transfer from fluid per unit total volume)
D_m	$[L^2/s]$	apparent molecular diffusivity of solute in solution in a porous medium including tortuosity effects, ($D_m \sim 1.\times 10^{-9}$ $[m^2/s]$ for NaCl at 20.°C).
$\underline{\underline{I}}$	[1]	identity tensor (ones on diagonal, zero elsewhere) (2x2)
$\underline{\underline{D}}(x,y,t)$	$[L^2/s]$	dispersion tensor
$\Gamma_w(x,y,t)$	$[M_s/M\cdot s]$	solute mass source in fluid (per unit fluid mass) due to production reactions

$C^*(x,y,t)$	$[M_s/M]$	solute concentration of fluid sources (mass fraction)
$C_s(x,y,t)$	$[M_s/M_G]$	specific concentration of adsorbate on solid grains (mass adsorbate/(mass solid grains plus adsorbate))
ρ_s	$[M_G/L_G^3]$	density of solid grains in solid matrix
$\Gamma_s(x,y,t)$	$[M_s/M_G \cdot s]$	adsorbate mass source (per unit solid matrix mass) due to production reactions within adsorbed material itself.

where $[L_G^3]$ is the volume of solid grains.

Equation (2.29) is the solute mass balance in terms of the dissolved mass fraction (solute concentration), C. The time derivative expresses the total changes in solute mass with time in a volume due to the mechanisms represented by terms on the right side of the equation. The term involving f(x,y,t) represents the loss of solute mass from solution which becomes fixed on the solid grain surfaces as adsorbate. The adsorbate source, f, may, in general, depend on solute concentration, C, adsorbate concentration, C_s, and the rate of change of these concentrations, depending on either an equilibrium adsorption isotherm or on non-equilibrium adsorption processes. SUTRA algorithms are structured to directly accept non-equilibrium sorption models as an addition to the code. However, the current version of SUTRA assumes equilibrium sorption as shown in the following section, "Adsorption and production/decay processes."

The term involving fluid velocity, $\underline{v}$, represents average advection of solute mass into or out of the local volume. The term involving molecular diffusivity of solute, D_m, and dispersivity, $\underline{\underline{D}}$, expresses the contribution of solute diffusion and dispersion to the local changes in solute mass. The diffusion contribution is based on a true physical process often negligible at the field

scale. The dispersion contribution is an approximation of the effect of solute advection and mixing in irregular flows which are not accounted for by solute advected by the average velocity. The solute mass source term involving $\Gamma_w(x,y,t)$, the solute mass production rate per unit mass of fluid, expresses the contribution to dissolved species mass of chemical, biological or radioactive reactions in the fluid. The last term accounts for dissolved species mass added by a fluid source with concentration, C*.

Equation (2.30) is the balance of mass which has been adsorbed by solid grain surfaces in terms of species concentration on the solid (specific adsorbate concentation), C_s. The change in total adsorbate mass is expressed by the time derivative term. It may increase due to species leaving solution as expressed by adsorbate source term, f. The adsorbed mass may also change due to a production of adsorbate mass (per unit solid matrix mass), Γ_s by radioactive or chemical processes within the adsorbate. Note that mass becomes immobile once adsorbed, and is affected only by possible desorption or chemical and biological processes.

The total mass of a species in a volume is given by the sum of solute mass and adsorbate mass. A balance of the total mass of a species is obtained by addition of (2.30) and (2.29). The general form of the total species mass balance used in SUTRA is this:

$$\frac{\partial(\varepsilon S_w \rho C)}{\partial t} + \frac{\partial[(1-\varepsilon)\rho_s C_s]}{\partial t} = -\underline{\nabla}\cdot(\varepsilon S_w \rho \underline{v} C)$$
$$+ \underline{\nabla}\cdot[\varepsilon S_w \rho(D_m \underline{\underline{I}} + \underline{\underline{D}})\cdot\underline{\nabla}C] + \varepsilon S_w \rho \Gamma_w + (1-\varepsilon)\rho_s \Gamma_s + Q_p C^* \qquad (2.31)$$

Equation (2.31) is the basis for SUTRA solute transport simulation. In cases of solute transport where adsorption does not occur ($C_s = 0$), the adsorbate source term, f, simply has the value zero ($f = 0$), and the terms that stem from equation (2.30) are ignored. Further discussion of solute and adsorbate mass balances may be found in Bear (1979).

Adsorption and production/decay processes

The volumetric adsorbate source, f, of (2.29) and (2.30) may be expressed in the terms of a specific sorption rate, f_s, as:

$$f = (1-\varepsilon)\rho_s f_s \qquad (2.32a)$$

$f_s(x,y,t)$	$[M_s/M_G \cdot s]$	specific solute mass adsorption rate (per unit mass solid matrix)

A particular non-equilibrium (kinetic) model of sorption is obtained by defining the functional dependence of the sorption rate, f_s, on other parameters of the system. For example, for a linear reversible non-equilibrium sorption model, the expression is: $f_s = m_1(C - m_2 C_s)$, where m_1 and m_2 are sorption parameters. This particular model and a number of other non-equilibrium sorption models are accommodated by a general expression for f_s, as follows:

$$f_s = \kappa_1 \frac{\partial C}{\partial t} + \kappa_2 C + \kappa_3 \qquad (2.32b)$$

where: $\kappa_1 = \kappa_1(C,C_s)$, $\kappa_2 = \kappa_2(C,C_s)$, $\kappa_3 = \kappa_3(C,C_s)$.

$\kappa_1(C,C_s)$	$[M/M_G]$	first general sorption coefficient
$\kappa_2(C,C_s)$	$[M/M_G \cdot s]$	second general sorption coefficient
$\kappa_3(C,C_s)$	$[M_s/M_G \cdot s]$	third general sorption coefficient

Through a suitable definition of the general coefficients, $\kappa_i(C,C_s)$, a number of non-equilibrium sorption models may be obtained. For example, the linear reversible non-equilibrium model mentioned above requires the definitions: $\kappa_1 \equiv 0$, $\kappa_2 \equiv m_1$, and $\kappa_3 \equiv -m_1 m_2 C_s$. The general coefficients κ_1, κ_2, and κ_3 are included in the SUTRA code to provide generality for possible inclusion of such non-equilibrium (kinetic) sorption models.

The equilibrium sorption models are based on definition of the general coefficients through the following relation:

$$\frac{\partial C_s}{\partial t} = \kappa_1 \frac{\partial C}{\partial t} \tag{2.33}$$

Only general sorption coefficient, κ_1, need be defined based on various equilibrium sorption isotherms as shown in the following. The other coefficients are set to zero, $\kappa_2 = \kappa_3 = 0$.

The linear equilibrium sorption model is based on the linear sorption isotherm assuming constant fluid density:

$$C_s = (\chi_1 \rho_o) C \tag{2.34a}$$

$$\frac{\partial C_s}{\partial t} = (\chi_1 \rho_o) \frac{\partial C}{\partial t} \tag{2.34b}$$

where:

χ_1 $\quad$ $|L_f^3/M_G|$ $\quad$ linear distribution coefficient

and ρ_o is the fluid base density

For linear sorption, general coefficient, κ_1, takes on the definition:

$$\kappa_1 = \chi_1 \rho_o \tag{2.34c}$$

The Freundlich equilibrium sorption model is based on the following isotherm which assumes a constant fluid density, ρ_o:

$$C_s = \chi_1(\rho_o C)^{\left(\frac{1}{\chi_2}\right)} \tag{2.35a}$$

$$\frac{\partial C_s}{\partial t} = \left(\frac{\chi_1}{\chi_2}\right)(\rho_o C)^{\left(\frac{1-\chi_2}{\chi_2}\right)} \rho_o \frac{\partial C}{\partial t} \tag{2.35b}$$

where:

χ_1	$\left[L_f^3/M_G\right]$	a Freundlich distribution coefficient
χ_2	[1]	Freundlich coefficient

when $\chi_2 = 1$, the Freundlich isotherm is equivalent to the linear isotherm. For Freundlich sorption, then, the general coefficient, κ_1, takes the definition:

$$\kappa_1 = \left(\frac{\chi_1}{\chi_2}\right) \rho_o^{\left(\frac{1}{\chi_2}\right)} C^{\left(\frac{1-\chi_2}{\chi_2}\right)} \tag{2.35c}$$

The Langmuir equilibrium sorption model is based on the following isotherm which assumes a constant fluid density, ρ_o:

$$C_s = \frac{\chi_1(\rho_o C)}{1 + \chi_2(\rho_o C)} \tag{2.36a}$$

$$\frac{\partial C_s}{\partial t} = \frac{\chi_1 \rho_o}{(1 + \chi_2 \rho_o C)^2} \frac{\partial C}{\partial t} \tag{2.36b}$$

where:

χ_1	$[L_f^3/M_G]$	a Langmuir distribution coefficient
χ_2	$[L_f^3/M_s]$	Langmuir coefficient

For very low solute concentrations, C, Langmuir sorption becomes linear sorption with linear distribution coefficient χ_1. For very high solute concentrations, C, the concentration of adsorbate mass, C_s, approaches an upper limit equal to (χ_1/χ_2). The general SUTRA coefficient, κ_1, is defined for Langmuir sorption as:

$$\kappa_1 = \frac{\chi_1 \rho_o}{(1 + \chi_2 \rho_o C)^2} \qquad (2.36c)$$

The production terms for solute, Γ_w, and adsorbate, Γ_s, allow for first-order mass production (or decay) such as linear BOD (biochemical oxygen demand) or radioactive decay, biological or chemical production, and zero-order mass production (or decay).

$$\Gamma_w = \gamma_1^w C + \gamma_o^w \qquad (2.37a)$$

$$\Gamma_s = \gamma_1^s C_s + \gamma_o^s \qquad (2.37b)$$

where:

γ^w_1	$[s^{-1}]$	first order mass production rate of solute
γ_o^w	$[(M_s/M)/s]$	zero-order solute mass production rate
γ_1^s	$[s^{-1}]$	first-order mass production rate of adsorbate
γ_o^s	$[(M_s/M_G)/s]$	zero-order adsorbate mass production rate

2.5 Description of Dispersion

Pseudo-transport mechanism

Dispersion is a pseudo-transport process representing mixing of fluids which actually travel through the solid matrix at velocities different from the average velocity in two space dimensions, $\underline{v}$, calculated from Darcy's law, (2.19). Dispersion is a pseudo-flux in that it only represents deviations from an average advective flux of energy or solute mass and as such does not represent a true mechanism of transport. Should it be possible to represent the true, complex, non-homogeneous velocity field in, for example, in the layers of an irregularly bedded field system, then the dispersion process need not be invoked to describe the transport, as the local variations in advection would provide the true picture of the transport taking place. However, as available data almost never allows for such a detailed velocity description, an approximate description, which helps to account for observed temperatures or concentrations different from that expected based on the average fluid advection, must be employed.

Current research trends are to develop dispersion models for various hydrogeological conditions, and SUTRA may be updated to include these new results as they become available. Currently, SUTRA dispersion is based on a new generalization for anisotropic media of the standard description for dispersion in isotropic homogeneous porous media. The standard description is, in fact, the only model available today for practical simulation. Because any inconsistencies which may arise in applying this dispersion model to particular

field situation often would not be apparent due to the poor quality or small amount of measured data, the user is warned to exercise good judgement in interpreting results when large amounts of so-called dispersion are required to explain the field measurements.

In any case, the user is advised to consult up-to-date literature on field-scale dispersion, before employing this transport model.

Isotropic-media dispersion model

The dispersion tensor, $\underline{\underline{D}}$, appearing in both energy and solute balances, (2.28) and (2.31), is usually expressed for flow in systems with isotropic permeability and isotropic spatial distribution of inhomogeneities in aquifer materials as:

$$\underline{\underline{D}} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \tag{2.38}$$

where, $\underline{\underline{D}}$ is, in fact, symmetric and the diagonal elements are:

$$D_{xx} = \left(\frac{1}{v^2}\right)(d_L v_x^2 + d_T v_y^2) \tag{2.39a}$$

$$D_{yy} = \left(\frac{1}{v^2}\right)(d_T v_x^2 + d_L v_y^2) \tag{2.39b}$$

and the off-diagonal elements are:

$$D_{ij} = \left(\frac{1}{v^2}\right)(d_L - d_T)(v_i v_j) \tag{2.39c}$$

$i \neq j, \quad i=x,y$

$\quad\quad\quad j=x,y$

$v(x,y,t)$	[L/s]	magnitude of velocity $\underline{v}$
$v_x(x,y,t)$	[L/s]	magnitude of x-component of $\underline{v}$
$v_y(x,y,t)$	[L/s]	magnitude of y-component of $\underline{v}$
$d_L(x,y,t)$	[L^2/s]	longitudinal dispersion coefficient
$d_T(x,y,t)$	[L^2/s]	transverse dispersion coefficient

The terms d_L and d_T [L^2/s] are called longitudinal and transverse dispersion coefficients, respectively. These terms are analogous to typical diffusion coefficients. What is special, is that these are directional in nature. The term, d_L, acts as a diffusion coefficient which causes dispersion forward and backward along the local direction of fluid flow, and is called the longitudinal dispersion coefficient. The term, d_T, acts as a diffusion coefficient causing dispersion evenly in the directions perpendicular to the local flow direction, and is called the transverse dispersion coefficient. Thus, if d_L and d_T were of equal value, a circular disk of tracer released (in the x-y plane) in ground water flowing, on the average uniformly and unidirectionally, would disperse in a perfectly symmetric circular manner as it moved downstream. However, if $d_L > d_T$ then the tracer would disperse in an elliptical manner with the long axis oriented in the flow direction, as it moved downstream.

The size of the dispersion coefficients are, in this model, for dispersion in isotropic permeability systems, dependent upon the absolute local magnitude of average velocity in a flowing system (Bear, 1979):

$$d_L = \alpha_L v \qquad (2.40a)$$

$$d_T = \alpha_T v \qquad (2.40b)$$

$\alpha_L(x,y)$	[L]	longitudinal dispersivity of solid matrix
$\alpha_T(x,y)$	[L]	transverse dispersivity of solid matrix

When the isotropic-media dispersion model is applied to a particular field situation where aquifer inhomogeneities are much smaller than the field transport scale, then dispersivities α_L and α_T may be considered to be fundamental transport properties of the system just as, for example, permeability is a fundamental property for flow through porous media. In cases where inhomogeneities are large or scales of transport vary, dispersivities may possibly not be representive of a fundamental system property. In this case, dispersion effects must be interpreted with care, because dispersivity values are the only means available to represent the dispersive characteristics of a given system to be simulated.

Anisotropic-media dispersion model

In a system with anisotropic permeability or anisotropic spatial distribution of inhomogeneities in aquifer materials, dispersivities may not have the same values for flows in all directions. In a case such as a layered aquifer, longitudinal dispersivity would clearly not have the same value for flows parallel to layers and perpendicular to layers. The isotropic-media dispersion model, described in the previous section, does not account for such variability as α_L is isotropic (direction-independent). Transverse dispersivity would also tend to be dependent on the flow-direction, but because it typically is only a small fraction of longitudinal dispersivity, especially in anisotropic media (Gelhar and Axness, 1983), its variability is ignored here. This does not imply that transverse dispersion is an unimportant process, but the approximation is made because accurate simulation of low transverse dispersion is already limited, due to the requirement of a fine mesh for accurate representation of the process. The effect of any direction-dependence of transverse dispersivity would be obscured by the numerical discretization errors in a typical mesh.

An ad-hoc model of flow-direction-dependent longitudinal dispersion is postulated. In this model, longitudinal dispersivity is assumed to have two principal directions (in two space dimensions) aligned with principal directions of permeability, x_p and x_m. The principal values of longitudinal dispersivity, are α_{Lmax} and α_{Lmin} in these principal directions (see Figure 2.4). Note that the subscripts, Lmax and Lmin, refer only to the maximum and minimum permeability directions, and are not intended to imply the relation in magnitude of α_{Lmax} and α_{Lmin}, the principal values of longitudinal dispersivity.

If F_s is the dispersive flux of solute (or energy) along a stream line of fluid flow, then

$$F_s = - \alpha_L \frac{\partial U}{\partial s} \tag{2.41}$$

where:

$\alpha_L(x,y,t)$	[L]	longitudinal dispersivity along a streamline

and U represents either concentration or temperature, and s is distance measured along a streamline. The dispersive flux components in the principal permeability directions x_p and x_m are:

$$F_p = - \alpha_{Lmax} \frac{\partial U}{\partial x_p} = F_s \cos \theta_{kv} \tag{2.42a}$$

$$F_m = - \alpha_{Lmin} \frac{\partial U}{\partial x_m} = F_s \sin \theta_{kv} \tag{2.42b}$$

where:

$\alpha_{Lmax}(x,y)$	[L]	Longitudinal dispersivity in the maximum permeability direction, x_p.
$\alpha_{Lmin}(x,y)$	[L]	Longitudinal dispersivity in the minimum permeability direction, x_m.
$\theta_{kv}(x,y,t)$	[°]	Angle from maximum permeability direction, x_p, to local flow direction, $(\underline{v}/\lvert\underline{v}\rvert)$

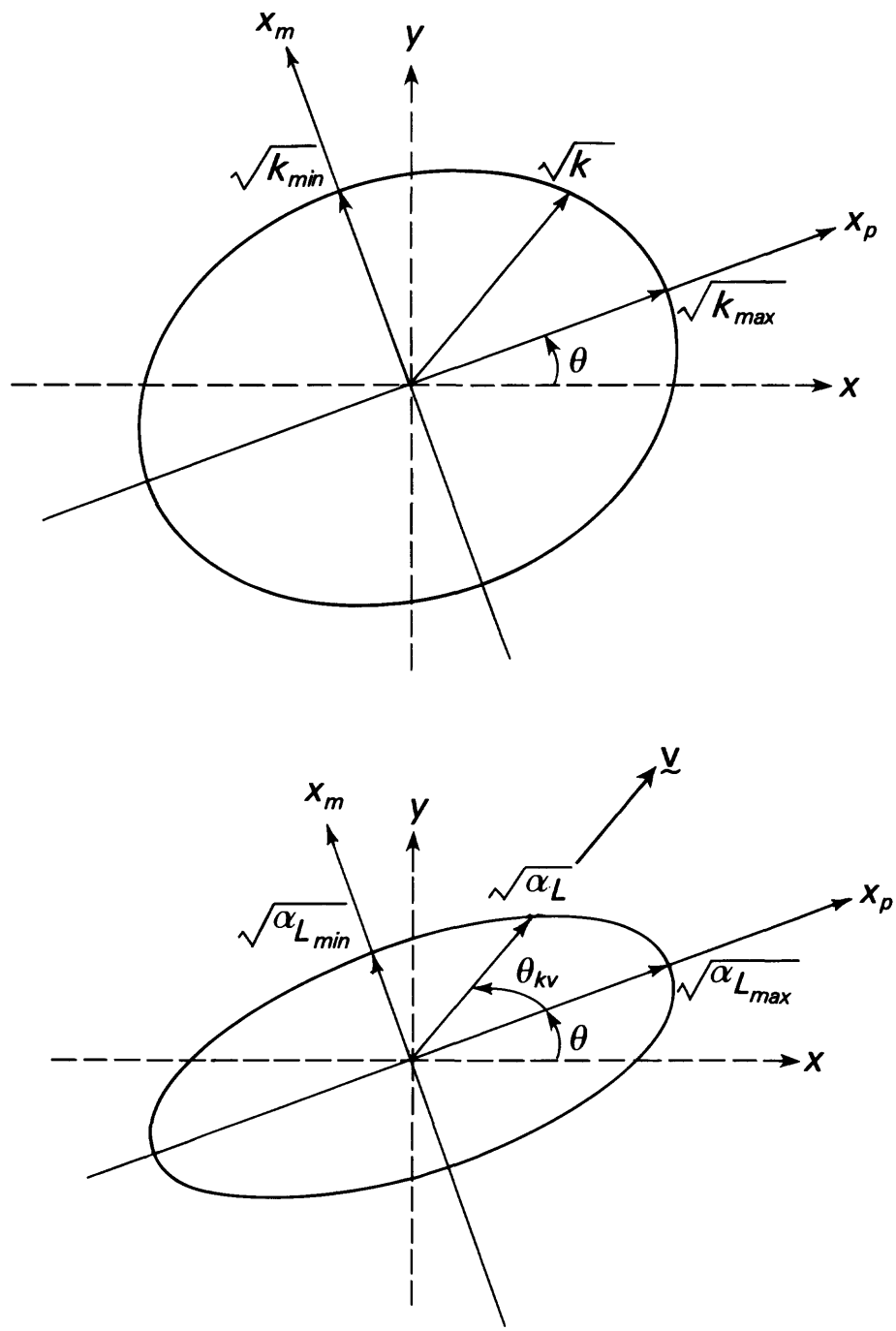


Figure 2.4
Definition of flow-direction-dependent longitudinal dispersivity, $\alpha_L(\theta)$.

Because U varies with x and y, U = U(x,y,t):

$$\frac{\partial U}{\partial s} = \frac{\partial U}{\partial x_p}\frac{\partial x_p}{\partial s} + \frac{\partial U}{\partial x_m}\frac{\partial x_m}{\partial s} \tag{2.43a}$$

$$\frac{\partial U}{\partial s} = \frac{\partial U}{\partial x_p}\cos\theta_{kv} + \frac{\partial U}{\partial x_m}\sin\theta_{kv} \tag{2.43b}$$

and:

$$F_s = -\alpha_L\left(\cos\theta_{kv}\frac{\partial U}{\partial x_p} + \sin\theta_{kv}\frac{\partial U}{\partial x_m}\right) \tag{2.44a}$$

$$F_s = \alpha_L\left[\cos^2\theta_{kv}\left(\frac{F_s}{\alpha_{Lmax}}\right) + \sin^2\theta_{kv}\left(\frac{F_s}{\alpha_{Lmin}}\right)\right] \tag{2.44b}$$

This defines an ellipse as:

$$\left(\frac{1}{\alpha_L}\right) = \left(\frac{\cos^2\theta_{kv}}{\alpha_{Lmax}}\right) + \left(\frac{\sin^2\theta_{kv}}{\alpha_{Lmin}}\right) \tag{2.45}$$

with semi-major axis $(\alpha_{Lmax})^{1/2}$ and semi-minor axis $(\alpha_{Lmin})^{1/2}$. The length of a radius is $(\alpha_L)^{1/2}$, as shown in Figure 2.4. This ellipse is analagous in concept to that which gives effective permeability in any direction in an anisotropic medium.

The value of effective longitudinal dispersivity as dependent on the flow direction is:

$$\alpha_L = \frac{\alpha_{Lmax}\,\alpha_{Lmin}}{\left(\alpha_{Lmin}\cos^2\theta_{kv} + \alpha_{Lmax}\sin^2\theta_{kv}\right)} \tag{2.46}$$

which is used by SUTRA to compute α_L for the anisotropic-media dispersion model. Note that if $\alpha_{Lmax} = \alpha_{Lmin}$, then the isotropic dispersion-media model is obtained.

This form of longitudinal dispersivity dependence on direction of flow relative to the principal permeability directions is similar to that obtained for a transversely isotropic medium in a stochastic analysis of macro-dispersion by Gelhar and Axness (1983).

Guidelines for applying dispersion model

Some informal guidelines may be given concerning values of dispersivities when other data are not available. Longitudinal dispersivities may be considered to be on the order of the same size as either the largest hydrogeologic or flow inhomogeneities along the transport reach or the distance between inhomogeneities, whichever is the greater value. For transport in pure homogeneous sand, longitudinal dispersivity is on the order of grain size. This is the type of situation where the isotropic-media dispersion model well describes observed transport behavior. In the case of a sandy aquifer containing well-distributed inclusions of less-permeable material, the longitudinal dispersivity required to correct an average advective transport which has passed by many of the inclusions would be of the order on the larger of either inclusion size or distance between inclusions.

Should the dispersivity, estimated on the basis of the size in homogeneities or distance between them, be greater than about one tenth of the longest transport reach, then the meaningful use of a constant-dispersivity dispersion model must be questioned. In such a case, the ideal action to take would be to more explicitly define the field distribution of velocity by taking into account the actual geometry of inhomogeneities. This would correctly account for most of the transport taking place as advective in nature, with much smaller con-

tributions of the approximate dispersive process. Given a better-defined velocity field, and in the absence of other data, dispersivity should then be chosen based on the largest postulated inhomogeneities met along a given average stream tube. The size and distribution of inhomogeneities not explicitly taken into account by the average flow field may be postulated based on the best available knowledge of local geology.

Transverse dispersivity, α_T, is typically even less well known for field problems than longitudinal dispersivity. Values of α_T used in simulation are typically between one tenth and one third of α_L. In systems with anisotropic permeability, α_T may be less than one hundredth of α_L for flows along the maximum permeability direction (Gelhar and Axness, 1983). Should simulated transport in a particular situation be sensitive to the value of transverse dispersivity, further data collection is necessary and the transport model must be interpreted with great care.

The ad-hoc model for longitudinal dispersion in anisotropic media presented in the previous section allows for simulation experiments with two principal longitudinal dispersivities which may be of special interest in systems with well-defined anisotropy values. Depending on the particular geometry of layers or inhomogeneities causing the permeability anisotropy, the longitudinal dispersivity in the minimum permeability direction, α_{Lmin}, may be either greater or smaller than that in the maximum permeability direction α_{Lmax}. However, use of the anisotropic-media dispersion model is advised only when clearly required by field data, and the additional longitudinal dispersion parameter is not intended for general application without evaluation of its applicability in a particular case.

2.6 Unified Description of Energy and Solute Transport

Unified energy-solute balance

The saturated-unsaturated ground-water energy balance (2.28) is simply an accounting of energy fluxes, sources and sinks which keeps track of how the energy per unit volume of solid matrix plus fluid, $[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s]T$, changes with time at each point in space. The saturated-unsaturated ground-water balance of solute plus adsorbate mass, (2.31), is similarly an accounting of solute and adsorbate fluxes, sources and sinks, which keeps track of how the species mass (solute plus adsorbate mass) per unit volume of solid matrix plus fluid, $(\varepsilon S_w \rho C + (1-\varepsilon)\rho_s C_s)$, changes with time at each point in space. Both balances, therefore, track a particular quantity per unit volume of solid matrix plus fluid.

The fluxes of energy and solute mass in solution, moreover, are caused by similar mechanisms. Both quantities undergo advection based on average flow velocity, $\underline{v}$. Both quantities undergo dispersion. Both quantities undergo diffusion; the diffusive solute mass flux is caused by molecular or ionic diffusion within the fluid, while the diffusive energy flux occurs by thermal conduction through both fluid and solid. Fluid sources and sinks give rise to similar sources and sinks of energy and solute mass. Energy and species mass may both be produced by zero-order processes, wherein energy may be produced by an endothermic reaction and solute may be produced, for example, by a biological process. The linear adsorption process affecting solutes is similar to the storage of energy in solid portion of an aquifer. Only the non-linear sorption processes and first-order production of solute and adsorbate, have no readily apparent analogy in terms of energy.

Thus, the balances of energy per unit volume, (2.28), and total species mass per unit volume, (2.31), may be expressed in a single unified balance in terms of a variable, U(x,y,t), which may either represent T(x,y,t) or C(x,y,t), as follows:

$$\frac{\partial}{\partial t}\left(\varepsilon S_w \rho c_w U\right) + \frac{\partial}{\partial t}\left[(1-\varepsilon)\rho_s U_s\right] + \underline{\nabla}\cdot\left(\varepsilon S_w \rho c_w \underline{v} U\right)$$

$$- \underline{\nabla}\cdot\left\{\rho c_w\left[\varepsilon S_w(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}) + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\}$$

$$= Q_p c_w U^* + \varepsilon S_w \rho \Gamma_w + (1-\varepsilon)\rho_s \Gamma_s \qquad (2.47)$$

where:

for energy transport

$$U \equiv T,\ U_s \equiv c_s T,\ U^* \equiv T^*,\ \sigma_w \equiv \frac{\lambda_w}{\rho c_w},\ \sigma_s \equiv \frac{\lambda_s}{\rho c_w} \qquad (2.47a)$$

$$\Gamma_w \equiv \gamma_o^w,\ \Gamma_s \equiv \gamma_o^s$$

for solute transport

$$U \equiv C,\ U_s \equiv C_s,\ U^* \equiv C^*,\ \sigma_w \equiv D_m,\ \sigma_s \equiv 0,\ c_w \equiv 1 \qquad (2.47b)$$

where C_s is defined by (2.34a), (2.35a) or (2.36a), depending on the isotherm.

By simple redefinition according to (2.47a) or (2.47b), equation (2.41) directly becomes the energy or species mass balance. This redefinition is automatically carried out by SUTRA as a result of whether the user specifies energy or solute transport simulation.

Fluid-mass-conservative energy-solute balance

A further consideration is required before obtaining the form of the unified energy/solute balance as implemented in SUTRA. The amount of energy or solute per unit combined matrix-fluid volume may change either due to a change in the total fluid mass in the volume even when concentration and temperature remain constant (see relation (2.10)). Such a change in fluid mass may be caused by changes in fluid saturation, or by pressure changes affecting compressive storage.

The energy and solute balances as well as their unified form, (2.47), track both types of contributions to changes in total stored energy or solute mass. However, the fluid saturation and pressure change contribution to energy and solute balances are already implicitly accounted for by the fluid mass balance.

The fluid mass balance contribution to solute and energy balances is expressed by the product of the fluid mass balance, equation (2.22) (which tracks changes in fluid mass per unit volume), with $c_w U$ (which represents either energy or solute mass per unit fluid mass). Note that $c_w = 1$ for solute transport. This product tracks energy or solute mass changes per unit volume due to fluid mass changes per unit volume:

$$(c_w U)\frac{\partial(\varepsilon S_w \rho)}{\partial t} + (c_w U)\,\underline{\nabla}\cdot(\varepsilon S_w \rho \underline{v}) = (c_w U)Q_p \qquad (2.48)$$

where the solute mass source, Γ, is neglected. Comparison of (2.48) with (2.47) will reveal that the terms on the left of (2.48) also appear in the unified balance equation.

Before substituting (2.48) for the duplicate terms in (2.47), the search for redundant terms may be extended to a balance of species mass or energy stored in the solid matrix rather than in the fluid. A simple mass balance for the solid matrix is:

$$\frac{\partial}{\partial t}\left[(1-\varepsilon)\rho_s\right] + \nabla\cdot\left[(1-\varepsilon)\rho_s \underline{v}_s\right] = 0 \qquad (2.49)$$

$\underline{v}_s$ [L/s] net solid matrix velocity

Due to the assumption that the net solid matrix velocity, $\underline{v}_s$, is negligable, the associated term of (2.49) is dropped. The contribution of this simple solid matrix mass balance to the unified solute-energy balance may again be obtained by taking the product of (2.49) with U_s:

$$(U_s)\frac{\partial}{\partial t}\left[(1-\varepsilon)\rho_s\right] = 0 \qquad (2.50)$$

A comparison reveals that this term also appears in (2.47).

The redundant information in the unified energy-solute balance which keeps track of both solid matrix and fluid mass balance contributions may be directly removed from (2.47) by subtracting (2.48) and (2.50). The result is:

$$\begin{aligned} &\varepsilon S_w \rho c_w \frac{\partial U}{\partial t} + (1-\varepsilon)\rho_s \frac{\partial U_s}{\partial t} + \varepsilon S_w \rho c_w \underline{v}\cdot\underline{\nabla}U \\ &- \underline{\nabla}\cdot\left\{\rho c_w\left[\varepsilon S_w(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}) + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\} \\ &= Q_p c_w (U^* - U) + \varepsilon S_w \rho \Gamma_w + (1-\varepsilon)\rho_s \Gamma_s \end{aligned} \qquad (2.51)$$

where:

for energy transport

$$U \equiv T,\ U_s \equiv c_s T\quad U^* \equiv T^*,\ \sigma_w \equiv \frac{\lambda_w}{\rho c_w},\ \sigma_s \equiv \frac{\lambda_s}{\rho c_w} \tag{2.51a}$$

$$\Gamma_w \equiv \gamma_o^w,\ \Gamma_s \equiv \gamma_o^s$$

for solute transport

$$U \equiv C,\ U_s \equiv C_s,\ U^* \equiv C^*,\ \sigma_w \equiv D_m,\ \sigma_s \equiv 0 \quad c_w \equiv 1 \tag{2.51b}$$

where C_s is defined by (2.34a), (2.35a) or (2.36a), depending on isotherm.

It is assumed in equation (2.51) that c_w and c_s are not time-dependent.

For numerical simulation, this equation may be termed a 'fluid-mass-conservative' form of the energy or species mass balance. When approximated numerically, the unified balance in the original form, (2.47), would contain approximation errors in both the fluid mass balance contributions (based on pressure and saturation changes) and the temperature or concentration change contribution. However, in the revised form, equation (2.51), the complete fluid mass balance contribution has already been analytically accounted for before any numerical approximation takes place. Thus, the total approximation error for the unified balance, (2.51), is significantly less as it is due to the temperature or concentration change contribution only.

The unified energy-species mass balance is brought to its final form by noticing that the form of the term, $\partial U_s/\partial t$, for energy transport, is the same as that for solute transport when using the equilibrium sorption relation (2.33), and that the form of the energy production of terms is similar to that of relations (2.37a) and (2.37b) for the mass production process:

$$\left[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s\right]\frac{\partial U}{\partial t} + \varepsilon S_w \rho c_w \underline{v}\cdot\underline{\nabla}U$$

$$- \underline{\nabla}\cdot\left\{\rho c_w\left[\varepsilon S_w(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}) + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\}$$

$$= Q_p c_w (U^* - U) + \varepsilon S_w \rho \gamma_1^w U + (1-\varepsilon)\rho_s \gamma_1^s U_s + \varepsilon S_w \rho \gamma_o^w + (1-\varepsilon)\rho_s \gamma_o^s \qquad (2.52)$$

where:

for energy transport

$$U \equiv T,\ U^* \equiv T^*,\quad \sigma_w \equiv \frac{\lambda_w}{\rho c_w},\ \sigma_s \equiv \frac{\lambda_s}{\rho c_w},\ \gamma_1^w \equiv \gamma_1^s \equiv 0 \qquad (2.52a)$$

for solute transport

$$U \equiv C,\ U_s \equiv C_s,\ U^* \equiv C^*,\ \sigma_w \equiv D_m,\ \sigma_s \equiv 0,\ c_s \equiv \kappa_1,\ c_w \equiv 1 \qquad (2.52b)$$

where C_s is defined by (2.34a), 2.35a) or (2.36a), and κ_1 is defined by (2.34c), (2.35c) or (2.36c), depending on the isotherm.

The fluid-mass-conservative form of the unified energy-species mass balance, (2.52), is exactly that which is implemented in SUTRA.

(p. 63 follows)

Chapter 3

Fundamentals of Numerical Algorithms

SUTRA methodology is complex because: (1) density-dependent flow and transport requires two interconnected simulation models, (2) fluid properties are dependent on local values of temperature or concentration, (3) geometry of a field area and distributions of hydrogeologic parameters may be complex, and (4) hydrologic stresses on the system may be distributed in space and change with time. Furthermore, a tremendous amount of data must be evaluated by SUTRA with precision. This requires great computational effort, and considerable numerical intricacy is required to minimize this effort. The mathematically elegant finite-element and integrated-finite-difference hybrid method employed by SUTRA allows great numerical flexibility in describing processes and characteristics of flow and transport in hydrologic field systems. Unlike simulation models based purely on the method of finite differences, however, the numerical aspects of which allow straight-forward interpretation at an intuitive level, some finite-element aspects of SUTRA methodology require interpretation at a less physical level and from a more mathematical point of view.

The following description of SUTRA numerical methods uses a simplified, constant-density water-table aquifer case as an illustrative example. While precise mathematically, this example is not used to demonstrate an actual application of SUTRA, as SUTRA does not, in fact, simulate a moving water table. The example is only used as a device through which to explain the theory and use of the primary numerical methods employed in SUTRA and the water table is invoked to allow discussion of a simple non-linearity. The basic methods, which are only demonstrated here, are applied in detail in

Chapter 4, "Numerical Methods," to the SUTRA fluid mass balance and unified energy-species mass balance.

The water-table aquifer fluid mass balance equation is useful for demonstration of basic numerical methods employed on SUTRA governing equations, because it displays some of the salient aspects of the SUTRA equations: a time derivative, a non-linear term involving space-derivatives, and a source term. The simplified fluid mass balance equation is as follows:

$$S_o \frac{\partial h}{\partial t} - \underline{\nabla}\cdot(K\underline{\nabla}h) = Q^* \qquad (3.1)$$

where

$$Q^* \equiv (Q_p/\rho)$$

and

S_o (x,y)	$[L^{-1}]$	specific storativity
h(x,y,t)	[L]	hydraulic head (sum of pressure head and elevation head)
K(x,y)	[L/s]	hydraulic conductivity (assumed for this example to be isotropic)
Q^*(x,y)	$[s^{-1}]$	volumetric fluid source (volume fluid injected per time / volume aquifer) (assumed constant for this example)
Q_p(x,y)	$[M/(L^3 \cdot s)]$	fluid mass source (mass fluid injected per time / volume aquifer) (assumed constant for this example)
ρ	$[M/L^3]$	fluid density (assumed constant for this example)

This equation, (3.1), is obtained from the SUTRA fluid mass balance, (2.24), by assuming saturated conditions, constant concentration and temperature, constant fluid density, and using the definition of hydraulic conductivity, $K \equiv (k\rho|\underline{g}|)/\mu$, where $|\underline{g}|$ is the acceleration of gravity, and of hydraulic

head, $h \equiv h_p$ + ELEVATION, where pressure head, $h_p \equiv p/(\rho|g|)$. For clarity, hydraulic conductivity is assumed to be isotropic in this example. While (3.1) may be considered a fully three-dimensional mass balance equation, it is assumed that flow takes place only areally in a water-table aquifer with a fixed impermeable base (at z-position, BASE(x,y)), and a moveable free surface (at z-position, h(x,y,t)). The z-direction is oriented vertically upward and the fluid is assumed to be in vertical hydrostatic equilibrium at any (x,y) position (no vertical flow). Aquifer thickness, B(x,y,t) [L], is measured as the distance along z from the free surface to the aquifer base, and may change with time. Aquifer transmissivity, $\mathcal{T}(x,y,t)$, is given by:

$$\mathcal{T} \equiv KB \equiv K(h - BASE) \qquad (3.2)$$

$\mathcal{T}(x,y,t)$	$[L^2/s]$	aquifer transmissivity
B(x,y,t)	[L]	aquifer thickness
BASE(x,y)	[L]	elevation of aquifer base

The above assumption, in effect, makes (3.1) a two-dimensional mass balance equation which is applied to a finite thickness aquifer. The two-dimensional form of (3.1) describing an areal fluid mass balance for water-table aquifers in terms of a head-dependent transmissivity arises during the basic numerical analysis of (3.1) in section 3.3, "Integration of Governing Equation in Space."

3.1 Spatial Discretization by Finite Elements

Although SUTRA is a two-dimensional model, the region of space in which flow and transport is to be simulated may be defined in three space dimensions. The three-dimensional bounded volume of an aquifer which is to be simulated by

SUTRA is completely divided up into a single layer of contiguous blocks. These blocks are called 'finite elements.' The subdivision is not done simply in a manner which creates one block (element) for each portion of the aquifer system which has unique hydrogeological characteristics. Each hydrogeologic unit is in fact divided into many elements giving the subdivided aquifer region the appearance of a fine net or mesh. Thus, subdivision of the aquifer region to be simulated into blocks is referred to as 'creating the finite-element mesh (or finite-element net).

The basic building block of a finite-element mesh is a finite element. The type of element employed by SUTRA for two-dimensional simulation is a quadrilateral which has a finite thickness in the third space dimension. This type of a quadrilateral element and a typical two-dimensional mesh is shown in Figure 3.1.

All twelve edges of the two-dimensional quadrilateral element are perfectly straight. Four of these edges are parallel to the z-coordinate direction. The x-y plane (which contains the two coordinate directions of interest) bisects each of the edges parallel to z, so that the top and bottom surfaces of the element are mirror images of each other reflected about the central x-y plane in the element. The mid-point of each z-edge (the point where the x-y plane intersects) is referred to as a nodal point (or node). Thus, the element has a three-dimensional shape, but always has only exactly four nodes, each of which in fact, represents the entire z-edge on which it is located. The nodes mark the fact that, in this type of element, some aquifer parameters may be assigned a different value at each z-edge of the element. The lack of nodes outside of the x-y plane is what makes this element two-dimensional; while some aquifer parameters may vary in value from node to node (i.e. from z-edge to z-edge), no parameters may be assigned varying values in the z-direction.

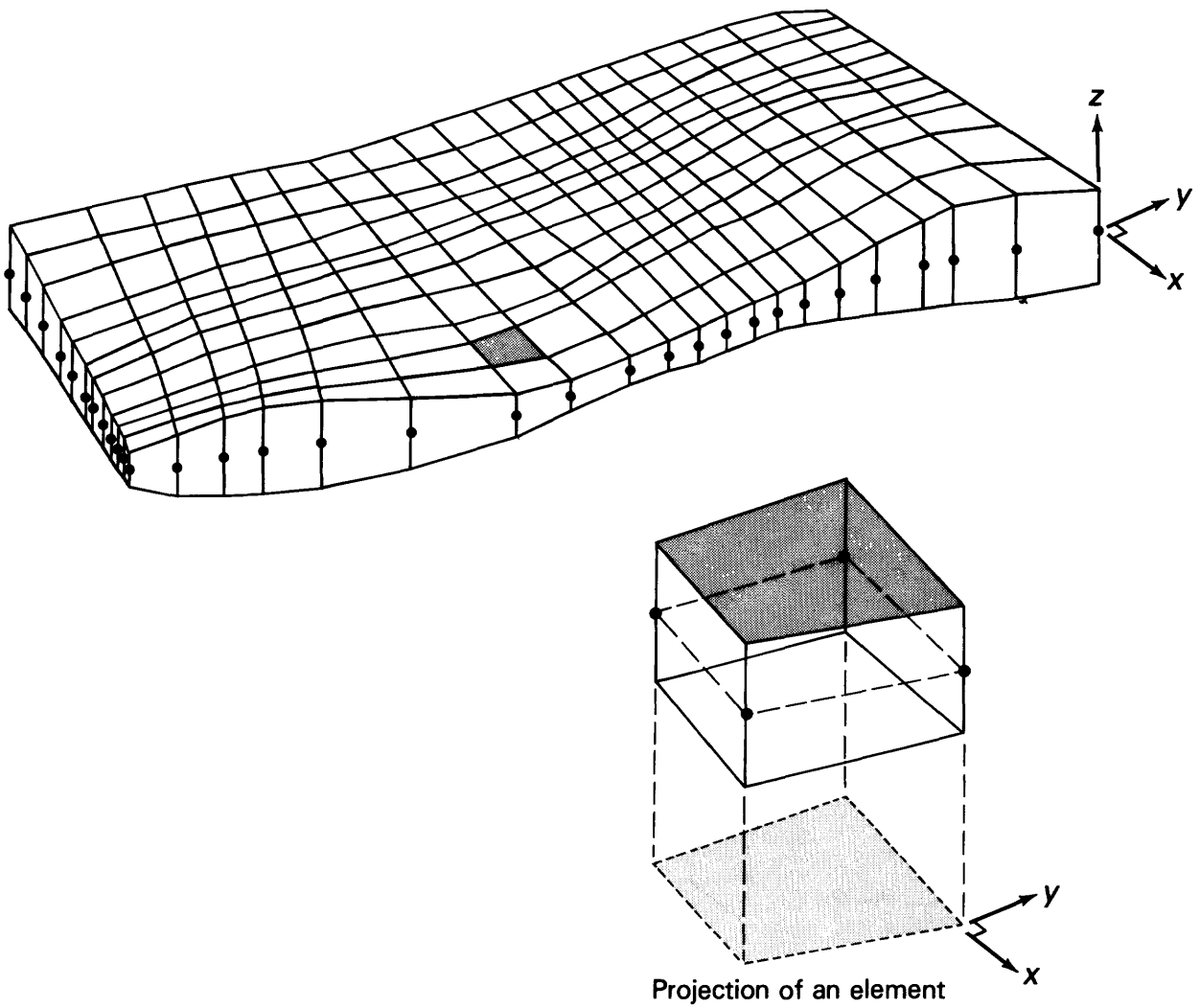


Figure 3.1
Two-dimensional finite-element mesh and quadrilateral element.

Within a two-dimensional finite-element mesh there is only a single layer of elements, the nodes of which lie in the x-y plane. Nodal points are always shared by the elements adjoining the node. Only nodes at external corners of the mesh are not contained in more than one element. The top and bottom surfaces are at every (x,y) point equidistant from the x-y plane, but the thickness of the mesh, measured in the z-direction, may vary smoothly from point to point. When projected on the x-y plane, as in Figure 3.1, a finite-element mesh composed of the type of elements used by SUTRA appears as a mesh of contiguous quadrilaterals with nodes at the corners. Hence, the term, 'quadrilateral element'.

3.2 Representation of Coefficients in Space

Aquifer parameters and coefficients which vary from point to point in an aquifer such as specific storativity, S_o, and hydraulic conductivity, K, are represented in an approximate way in SUTRA. Parameters are either assigned a particular constant value in each element of a finite-element mesh (elementwise), or are assigned a particular value at each node in the mesh in two possible ways (nodewise or cellwise).

In the water-table aquifer, for a simple example, a regular two-dimensional mesh is used. The steplike appearance of elementwise assignment of K values over this simple mesh is shown in Figure 3.2. Nodewise assignment for head over this mesh results in a continuous surface of h values as shown in Figure 3.3, with linear change in value between adjoining nodes along (projected) element edges. Cellwise assignment is employed for specific storativity, S_o, and the time derivative, $\frac{\partial h}{\partial t}$. This results in a steplike appearance of the assigned values

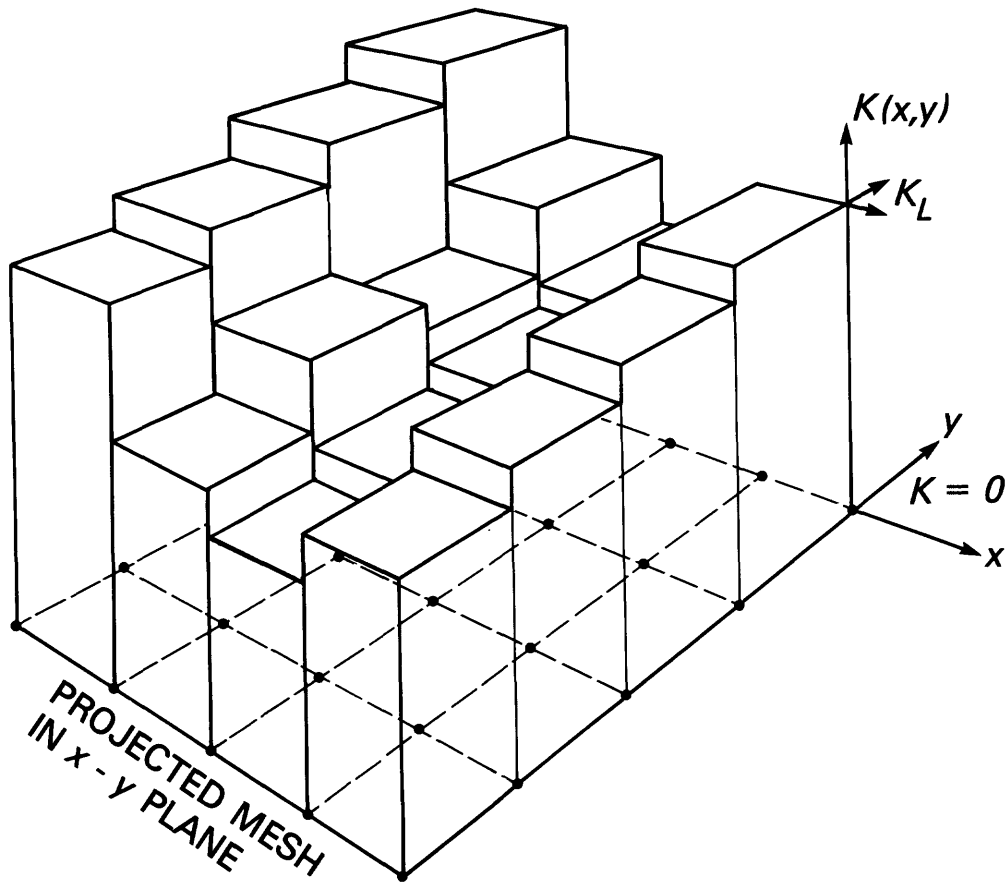


Figure 3.2
Elementwise discretization of coefficient K(x,y).

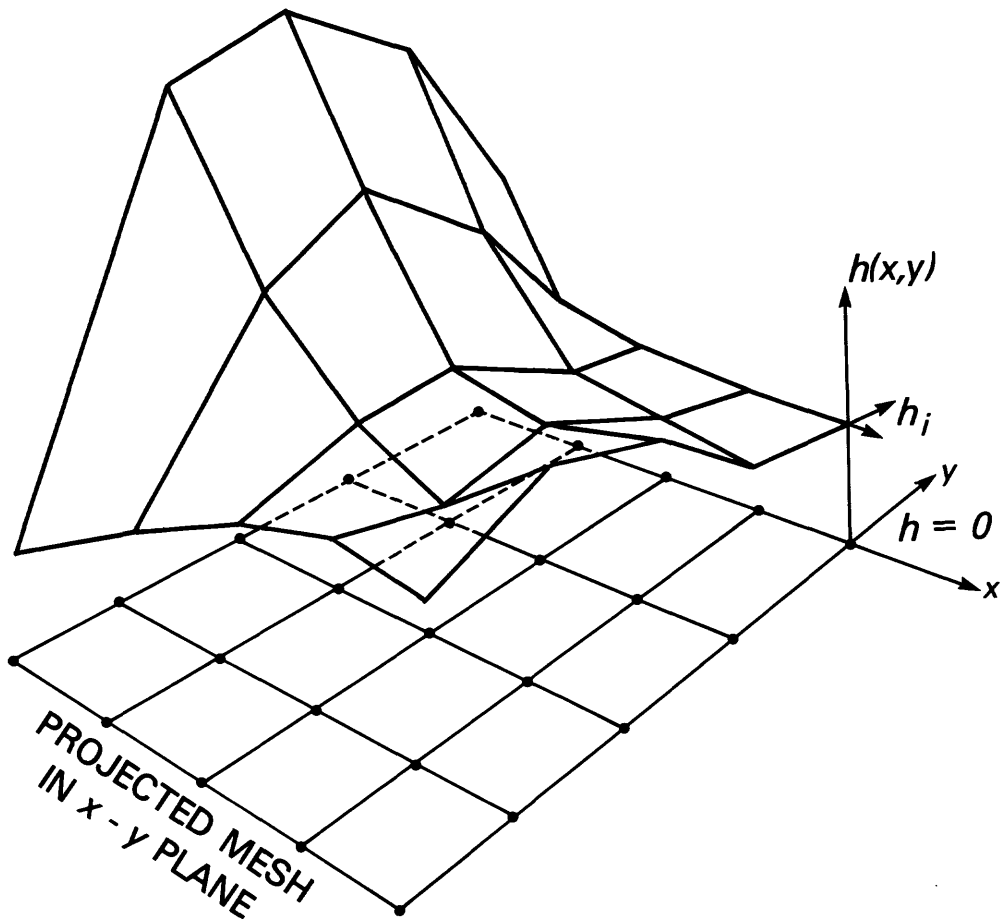


Figure 3.3
Nodewise discretization of coefficient h(x,y).

over the mesh similar to that of elementwise assignment in Figure 3.2, but each cell is centered on a node, not on an element. Cell boundaries are half way between opposite sides of an element and are shown for the regular mesh in Figure 3.4. Thus the spatial distributions of parameters, K, h and S, are discretized (i.e., assigned discrete values) in three different ways: K, elementwise, h, nodewise, and S_o, cellwise.

Because the internal program logic depends on the type of discretization, SUTRA expects certain particular parameters or equation terms to be discretized elementwise, nodewise, or cellwise. The primary dependent variables of the SUTRA code p, and T or C, (in this example case, only hydraulic head, h), are expressed nodewise when used in terms which calculate fluxes of fluid mass, solute mass or energy.

Elementwise discretization

The equation which gives the values, over the finite element mesh, of an elementwise parameter, may be expressed for the hydraulic conductivity of the present example as:

$$K(x,y) \approx \sum_{L=1}^{NE} K_L(x,y) \qquad (3.3)$$

where the elements have been numbered from one to NE (total number of elements in the mesh), and $K_L(x,y)$ [L/s] has the value of hydraulic conductivity of element L for (x,y) coordinates within the element, and a value of zero outside the element. Thus $K_L(x,y)$ is the flat-topped 'box' standing on an element L, in Figure 3.2, and K(x,y) is represented in a discrete approximate way by the sum of all the 'boxes'. Note that $K_L(x,y)$ has the same value in the z-direction from the top to the bottom of each two-dimensional element.

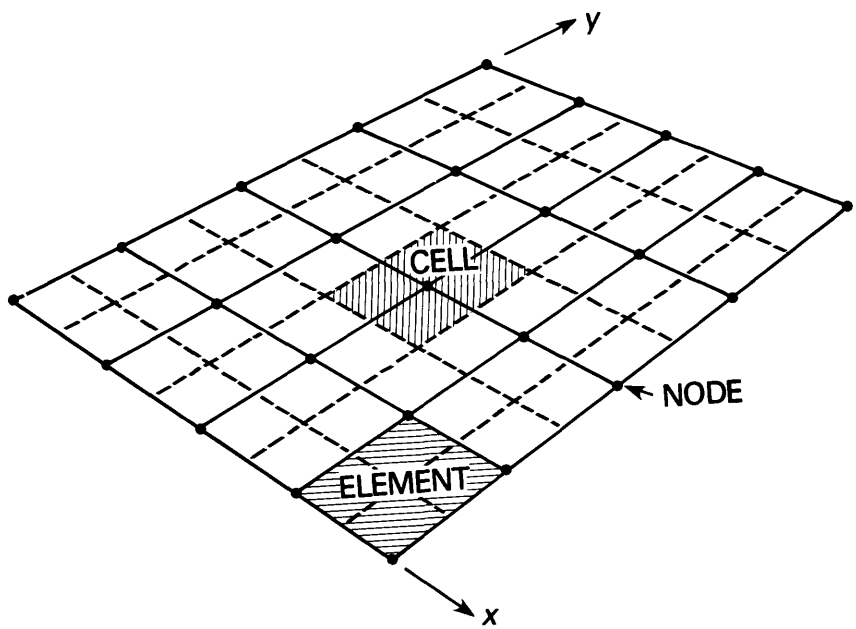


Figure 3.4
Cells, elements and nodes for a two-dimensional finite-element mesh composed of quadrilateral elements.

Nodewise discretization

The equation which gives the values, over the finite-element mesh, of a nodewise value, may be expressed for the two-dimensional mesh as:

$$h(x,y,t) \approx \sum_{j=1}^{NN} h_j(t)\ \phi_j(x,y) \qquad (3.4)$$

where the nodes have been numbered from one to NN (total number of nodes in the mesh). There are NN coefficients, $h_j(t)$, each of which is assigned the value of head at the coordinates (x_j,y_j) of node number, j. These nodal head values may change with time to represent transient responses of the system. The function, $\phi_j(x,y)$, is known as the 'basis function'. It is the basis functions which spread values of head between the nodes when head is defined only at the nodal points by values of h. There is one basis function $\phi_j(x,y)$ defined for each node, j, of the NN nodes in the mesh. Suffice it to say, at this point, that at the node j, to which it belongs, the basis $\phi_j(x,y)$, has a value of one. At all other nodes i, $i \neq j$, in the mesh, it has a value of zero. It drops linearly in value from one to zero along each projected element edge to which the node j is connected. This means that even when all the NN products of h_j and $\phi_j(x,y)$ are summed (as in relation (3.4)), if the sum is evaluated at the coordinates (x_j,y_j) of node j, then h(x,y) exactly takes on the assigned value, h_j. This is because the basis function belonging to node j has a value of one at node j, and all other basis functions belonging to other nodes, i, $i \neq j$, have a value zero at node j dropping them from the summation in (3.4). Basis functions are described mathematically in section 4.1, "Basis and Weighting Functions."

Cellwise discretization

The equation which gives the values, over the finite-element mesh, of a cellwise parameter may be expressed for the specific storativity of the present example as:

$$S_o(x,y) \approx \sum_{i=1}^{NN} S_i(x,y) \qquad (3.5)$$

where $S_i(x,y)$ has the value of specific storativity of the cell centered on node i for (x,y) coordinates within the cell, and a value of zero outside the cell. Thus, $S_i(x,y)$ is a flat topped 'box' standing on a cell i in Figure 3.4, and $S_o(x,y)$ is represented in a discrete approximate way by the sum of all the 'boxes'. Note $S_i(x,y)$ has the same value in the z-direction from the top to bottom of each two-dimensional element.

Reviewing the example problem, K is assigned elementwise and both S_o and $\frac{\partial h}{\partial t}$ are assigned cellwise. Hydrualic head, h(x,y,t), and element thickness, B(x,y,t), measured in the z-direction, are both discretized nodewise, with the nodewise expansion for thickness:

$$B(x,y) \approx \sum_{i-1}^{NN} B_i(t)\phi_i(x,y) \qquad (3.6)$$

The values $B_i(t)$ are the NN particular values which element thickness has at the nodes, and these values may change with time in the present water-table example. Relation (3.6) should call to mind a vision of discretized values of thickness represented by a surface similar to that of Figure 3.3. The head surface of Figure 3.3 may stretch or shrink to move up or down as the head values at nodes, $h_i(t)$, change with time due to stresses on the aquifer system. The

nodewise discretized surface may be viewed as the water table, and the element thickness as the thickness of the water-table aquifer.

3.3 Integration of Governing Equation in Space

Approximate governing equation and weighted residuals method

The governing equation for the water-table example may be written in operator form as:

$$O(h) = S_o \frac{\partial h}{\partial t} - \underline{\nabla}\cdot(K\underline{\nabla}h) - Q^* = 0 \qquad (3.7)$$

Certain variables in this equation are approximated through elementwise and nodewise discretization. Particular terms of the equation are approximated through cellwise discretization. The result is that neither the derivatives, nor the variables are described exactly. Relation (3.7) no longer exactly equals zero:

$$\hat{O}(h) = R(x,y,t) \qquad (3.8)$$

where $\hat{O}(h)$ is the result of approximating the terms of the equation and the variables, and $R(x,y,t)$ is the residual value of the approximated equation. When simulating a system with a numerical model based on approximation of the governing equation, $\hat{O}(h)$, the residual, R, must be kept small everywhere in the simulated region and for the entire time of simulation in order to accurately reproduce the physical behavior predicted by the exact governing equation, (3.7).

In order to achieve a minimum error, a method of weighted residuals is applied to (3.8). The purpose of the method of weighted residuals is to minimize the error of approximation in particular sub-regions of the spatial domain

to be simulated. This is done by forcing a weighted average of the residual to be zero over the sub-regions. This idea is the most abstract of those required to understand SUTRA methodology. The Galerkin method of weighted residuals chooses to use the 'basis function', $\phi_i(x,y)$, mentioned in the previous section, as the weighting function for calculation of the average residual:

$$\int_V \hat{O(h)}\ \phi_i(x,y)\ dV = \int_V R(x,y,t)\ \phi_i(x,y)\ dV = 0 \qquad (3.9)$$

$$i = \overline{1,NN}$$

where V is the volume of the region to be modeled. The model volume is completely filled by a single layer of quadrilateral finite elements. Relation (3.9) is actually NN relations, one for each of NN nodes in the finite element mesh as indicated by the notation, $i = \overline{1,NN}$.

In each relation, the integral sums the residual weighted by the basis function over a volume of space. Each integrated weighted residual is forced to zero over the region of space in which $\phi_i(x,y)$ is non-zero. This region includes only elements which contain node i, because of the manner in which the basis function is defined, as described earlier. Thus, over each of these NN sub-regions of a mesh, the sum of positive and negative residuals after weighting is forced to zero by relation (3.9). This, in effect, minimizes the average error in approximating the governing equation over each sub-region.

After stating that the integral of weighted residuals must be zero for each sub-region of the mesh as in (3.9), the derivation of the numerical methods becomes primarily a job of algebraic manipulation. The process is begun by substitution of the governing equation for $\hat{O(h)}$ in (3.9):

$$\int_V \left(S_o \widehat{\frac{\partial h}{\partial t}}\right) \phi_i(x,y)\, dV - \int_V \left(\underline{\nabla}\cdot(\widehat{K\underline{\nabla}h})\right) \phi_i(x,y) dV \qquad (3.10)$$

$$-\int_V \left(\widehat{Q^*}\right) \phi_i(x,y)\, dV = 0$$

$$i = \overline{1,NN}$$

The terms in large parentheses topped by a carat are the approximate discrete forms of the respective terms in (3.7). These are expanded in the manipulations that follow. Relation (3.10) is discussed term by term in the following paragraphs.

Cellwise integration of time-derivative term

The first term involving the volume integral of the time derivative may be written in terms of the three space dimensions, x, y, and z. Although the governing equation and parameters vary only in two space dimensions, they apply to the complete three-dimensional region to be modeled.

$$\int_V \left(S_o \widehat{\frac{\partial h}{\partial t}}\right) \phi_i(x,y)\, dV = \int_z \int_y \int_x \left(S_o \widehat{\frac{\partial h}{\partial t}}\right) \phi_i(x,y)\, dz\, dy\, dx \qquad (3.11)$$

$$= \int_y \int_x \left(S \widehat{\frac{\partial h}{\partial t}}\right) \phi_i(x,y) \left[\int_z dz\right] dx\, dy$$

The rearrangement in the final term of (3.11) is possible because no parameter depends on z. In fact, referring to (3.2), the aquifer thickness, B(x,y,t),

may be defined as:

$$B(x,y,t) = \int_{z(t)} dz = h(x,y,t) - BASE(x,y) \tag{3.12}$$

The final term of (3.11) is then:

$$\int_y \int_x \left(S_o \widehat{\frac{\partial h}{\partial t}} \right) \phi_i(x,y)\ B(x,y,t)\ dx\ dy \tag{3.13}$$

Now cellwise discretization is chosen for S_o and for $\frac{\partial h}{\partial t}$, making these terms take on a constant value for the region of each cell i. The region of cell i is the same region over which $S_i(x,y)$ is non-zero. Then, for any cell i, term (3.13) becomes:

$$S_i \frac{\partial h}{\partial t} i \int_y \int_x \phi_i(x,y)\ B(x,y,t)\ dx\ dy \tag{3.14}$$

where S_i and $\frac{\partial h}{\partial t} i$ are the values taken by S_o and $\frac{\partial h}{\partial t}$ in cell i.

It can be shown that the volume of cell i, denoted by $V_i(t)$, is, in fact, the integral in (3.14):

$$V_i(t) = \int_y \int_x \phi_i(x,y)\ B(x,y,t)\ dx\ dy \tag{3.15}$$

For a particular finite-element mesh, the volume $V_i(t)$ of each cell is determined by numerical integration of (3.15). Numerical integration by Gaussian quadrature is discussed in section (4.3), "Gaussian Integration."

Given the value of the specific storativity of each cell, S_i, the time derivative of head in each cell, $\frac{\partial h}{\partial t}i$, and given the volume of each cell, $V_i(t)$, determined numerically, the first term of the weighted residual statement takes on its discrete approximation in space:

$$\int_V \left(S_o \widehat{\frac{\partial h}{\partial t}}\right) \phi_i(x,y)\, dV = S_i \frac{\partial h}{\partial t}i\, V_i(t) \qquad (3.16)$$

Elementwise integration of flux term and origin of boundary fluxes

Manipulation of the second integral in (3.10) begins with the application of Green's theorem which is an expanded form of the divergence theorem. This converts the integral into two terms, one of which is evaluated only at the surface of the region to be simulated. Green's theorem is:

$$\int_V \left(\underline{\nabla}\cdot\underline{W}\right) A\, dV = \int_\Gamma \left(\underline{W}\cdot\underline{n}\right) A\, d\Gamma - \int_V \left(\underline{W}\cdot\underline{\nabla}A\right) dV \qquad (3.17)$$

where A is a scalar and $\underline{W}$ is a vector quantity. The boundary of volume V is denoted by Γ including both edges and upper and lower surfaces of the aquifer, and $\underline{n}$ is a unit outward normal vector to the boundary. Application of (3.17) to the second term in (3.10) results in:

$$-\int_V \left[\underline{\nabla}\cdot\left(K\widehat{\underline{\nabla}h}\right)\right] \phi_i(x,y)\, dV = -\int_\Gamma \left[\left(K\widehat{\underline{\nabla}h}\right)\cdot\underline{n}\right] \phi_i\, d\Gamma$$

$$+ \int_V \left(K\,\widehat{\underline{\nabla}h}\right)\cdot\underline{\nabla}\phi_i\, dV \qquad (3.18)$$

The first term on the right of (3.18) contains a fluid flux given by Darcy's law:

$$\varepsilon\, v_{OUT} = - K\underline{\nabla}h \cdot \underline{n} \tag{3.19}$$

where v_{OUT} is the outward velocity at the boundary normal to the bounding surface. Thus the integral gives the total flow out across the bounding surface, Q_{OUT_i}, in the vicinity of a node i on the surface:

$$Q_{OUT_i} = \int_{\Gamma} \left(\varepsilon v_{OUT}\, \phi_i \right) d\Gamma \tag{3.20}$$

An inflow would have a negative value of Q_{OUT_i}, and the relation between an inflow, Q_{IN_i}, and outflow is: $Q_{IN_i} = -Q_{OUT_i}$. Thus, the first integral on the right of (3.18) represents flows across boundaries of the water-table aquifer model.

The second integral on the right of (3.18) may be expressed in three spatial coordinates.

$$\begin{aligned} \int_V \left(K \hat{\underline{\nabla}} h \right) \cdot \underline{\nabla} \phi_i \, dV &= \int_x \int_y \int_z \left(K \hat{\underline{\nabla}} h \right) \cdot \underline{\nabla} \phi_i \, dz \, dy \, dx \\ &= \int_x \int_y \left(K \hat{\underline{\nabla}} h \right) \cdot \underline{\nabla} \phi_i \left[\int_z dz \right] dy \, dx = \int_x \int_y \left(K \hat{\underline{\nabla}} h \right) \cdot \underline{\nabla} \phi_i \, B(x,y,t) \, dy \, dx \end{aligned} \tag{3.21}$$

No term varies in the z-direction, allowing the use of (3.12) which defines aquifer thickness B. Notice that the transmissivity as given by (3.2), $T = KB$ appears in the form of the integral just obtained.

Now the approximation for the term $K \underline{\nabla} h$ is substituted into the integral. Hydraulic head, h(x,y,t), is approximated in a nodewise manner as given by relation (3.4). The integral of (3.2) becomes:

$$\int_x \int_y \left(K \underline{\nabla} h\right) \cdot \nabla \phi_i \; B \; dy \; dx = \int_x \int_y \left[\hat{K} \; \underline{\nabla} \sum_{j=1}^{NN} h_j(t) \; \phi_j(x,y)\right] \cdot \underline{\nabla} \phi_i \; B \; dy \; dx$$

$$= \sum_{j=1}^{NN} h_j(t) \int_x \int_y \hat{K} \left(\underline{\nabla} \phi_j \cdot \underline{\nabla} \phi_i\right) B \; dy \; dx = \sum_{j=1}^{NN} h_j(t) \; I_{ij}(t) \qquad (3.22)$$

where $\hat{K}$ is the elementwise approximation for K(x,y). The summation and $h_j(t)$ may be factored out of the integral because h_j is a value of head at a node and does not vary with x and y location. The integral is represented by $I_{ij}(t)$ and depends on time because aquifer thickness, B, is time-dependent. For each node i, there are apparently j=NN integrals which need to be evaluated. In fact, due to the way in which basis functions are defined, there are only a few which are non-zero, because $(\underline{\nabla} \phi_j \cdot \underline{\nabla} \phi_i)$ is non-zero only when nodes i and j are in the same finite element. When nodes i and j are in different elements, then $\underline{\nabla} \phi_j$ is zero in the element containing node i.

The integrals are evaluated numerically by Gaussian integration. This is accomplished by first breaking up the integral over the whole volume to be simulated, into a sum of integrals, one each over every finite element in the mesh:

$$I_{ij}(t) = \int_x \int_y \hat{K} \left(\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i \right) B \, dy dx = \sum_{L=1}^{NE} \int_{x_L} \int_{y_L} \hat{K} \left(\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i \right) B \, dy \, dx \qquad (3.23)$$

There are NE elements in the mesh, L is the element number, and x_L and y_L are the x and y spatial domains of element L. Thus, for a given L, the integral over x_L and y_L is integrated only over the area of element L.

Now the discrete elementwise approximation for hydraulic conductivity, as given by (3.3) allows one term for element L in the summation of (3.23) to be written as:

$$K_L \int_{x_L} \int_{y_L} \left(\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i \right) B \, dy \, dx \qquad (3.24)$$

Here, the thickness B is specified to vary nodewise. The formula for B in this example is obtained by substituting the nodewise expression for head, (3.4), into the definition of B, relation (3.2).

The integral over one element, as given by term (3.24), must be evaluated numerically. In order to do this, the coordinates of the element L, which has an arbitrary quadrilateral shape as suggested in Figure 3.3, is transformed to a new coordinate system in which the element is a two-by-two square. Then, Gaussian integration is carried out to evaluate the integral. For a given combination of nodes i and j, this transformation and numerical integration is carried out for all elements in the mesh in which both nodes i and j appear. (There are 16 i-j combinations evaluated in each quadrilateral element.) The elementwise pieces of the integral for each i-j combination are then summed according to (3.23) in order to obtain the value of the integral over the whole

region. The summation is called the 'assembly' process. This element transformation, integration of the 16 integrals arising in each element, and summation, makes up a large part of the computational effort of a finite-element model and also requires the most complex algorithm in a finite-element model. It is in this way that the second term of (3.10) is evaluated. More information on finite-element integration and assembly may be found in numerical methods texts such as Wang and Anderson (1982), Pinder and Gray (1977), or Huyakorn and Pinder (1983). The details of this method as applied in SUTRA are given in Chapter 4, "Numerical Methods."

Cellwise integration of source term

The last term of (3.10) deals with sources of fluid to the aquifer such as injection wells. The volume integral may, as before, may be written in x,y, and z coordinates:

$$-\int_V Q^*(x,y)\ \phi_i(x,y)\ dV = -\int_x\int_y\int_z Q^*\ \phi_i\ dz\ dy\ dx$$

$$= -\int_x\int_y Q^*\ \phi_i\ B(x,y,t)\ dy\ dx \qquad (3.25)$$

where thickness B is introduced because Q^* and ϕ_i do not vary with z. It is assumed that all fluid entering the aquifer within the region of cell i, which surrounds node i, enters at node i. If Q_i^* [L^3/s] is defined as the volume of fluid entering cell i per unit time, then Q^* [s^{-1}], which is the volume of fluid entering the aquifer per unit volume aquifer per unit time, is given as:

$$Q^*(x,y) = \sum_{i=1}^{NN} \left(\frac{Q^*_i}{V_i}\right) \tag{3.26}$$

This is a cellwise discretization for the source term, Q^*. For cell i:

$$-\int_x \int_y Q^* \phi_i \, B \, dy \, dx = -\left(\frac{Q^*_i}{V_i}\right) \int_x \int_y \phi_i \, B \, dy \, dx = -Q^*_i \tag{3.27}$$

Thus all recharges within cell i due to areal infiltration, well injection or other types are allocated to the source at node i.

This completes the spatial integration of the governing equation for the example problem.

3.4 Time Discretization of Governing Equation

When the integrated terms of the governing equation are substituted in (3.10) the following results:

$$S_i \, V_i(t) \frac{dh_i}{dt} + \sum_{j=1}^{NN} I_{ij}(t) \, h_j(t) = Q_{IN_i} + Q^*_i \tag{3.28}$$

$$i = \overline{1,NN}$$

These are NN integrated weighted residual approximations of the governing differential equation, one at each node i in the mesh. Because of the summation term in (3.28), the integrated approximate equation for a node, i, may involve the values of head, $h_j(t)$, at all other nodes in the mesh. The other terms in (3.28) involve only values at node i itself, at which the entire relation is evaluated.

All the parameters in (3.28) are no longer functions of the space coordinates. Each parameter takes on a particular value at each node in the mesh. Some of these values vary with time and a particular time for evaluation of these values needs to be specified. Also, the time derivative requires discretization.

Time steps

Time is broken up into a series of discrete steps, or time steps. The length of a time step, Δt, is the difference in time between two discrete times, at the beginning and end of a time step:

$$\Delta t_{n+1} = t^{n+1} - t^{n} \tag{3.29}$$

where Δt_{n+1} is the length of the $(n+1)^{th}$ time step, t^n is the actual time at the beginning of the $(n+1)^{th}$ time step and t^{n+1} is the actual time at the end of this time step. The time steps are chosen to discretize the time domain before a simulation just as a mesh (or 'spatial steps') is chosen to discretize space. The time step length may vary from step to step.

The entire spatially integrated governing equation, (3.28), is evaluated at the end of each time step, $t = t^{n+1}$. The time derivative of head in (2.28) is approximated, using a finite-difference approximation, as the change in head over a time step, divided by the time step length:

$$\frac{dh_i}{dt} \simeq \frac{h_i(t^n + \Delta t_{n+1}) - h_i(t^n)}{\Delta t_{n+1}} \tag{3.30}$$

In order to simplify the notation, the head at the end of the time step,

$h_i(t^n + \Delta t_{n+1})$ is denoted h_i^{n+1} , and the head at the beginning of the time step $h_i(t^n)$ is denoted h_i^n. Thus,

$$\frac{dh_i}{dt} \simeq \frac{h_i^{n+1} - h_i^n}{\Delta t_{n+1}} \tag{3.31}$$

The parameters that depend on time in (3.28), $V_i(t)$, $I_{ij}(t)$ and $h_j(t)$, are also evaluated at the time, t^{n+1}, at the end of a time step:

$$h_j(t)\Big|_{t^{n+1}} = h_j^{n+1} \tag{3.32a}$$

$$V_i(t)\Big|_{t^{n+1}} = V_i^{n+1} \tag{3.32b}$$

$$I_{ij}(t)\Big|_{t^{n+1}} = I_{ij}^{n+1} \tag{3.32c}$$

The sources, Q_{IN_i}, and Q_i^*, are assumed constant in time for present example.

Resolution of non-linearities

The variability in time of cell volume, V_i, and the integral, I_{ij}, depends on the changing thickness of the aquifer with time, B(x,y,t). The aquifer thickness at node i at the end of a time step, B_i^{n+1}, is not known until the head at the end of the time step is known giving the water-table elevation. This typifies a non-linear problem wherein the problem requires values of coefficients in order to be solved, but the values of these coefficients depend on the, as yet unobtained solution. This circular problem is avoided in this example by using estimates of the coefficient values in the solution. An estimate of the head at the end of the next time step is obtained by a linear projection:

$$h_i^{proj} = h_i^n + \left(\frac{\Delta t_{n+1}}{\Delta t_n}\right)\left(h_i^n - h_i^{n-1}\right) \tag{3.33}$$

where h_i^{proj} is the projected or estimated head at the end of the, as yet unsolved time step, which would have an exact value, h_i^{n+1}. Actually, in addition to projection, SUTRA also employs a simple iterative process to resolve nonlinearities. This is described in sections 4.4 and 4.5 under the sub-heading "Temporal discretization and iteration."

A projected thickness may then be determined from (3.33) as:

$$B_i^{n+1} \simeq B_i^{proj} = h_i^{proj} - BASE_i \qquad (3.34)$$

where B_i^{n+1} is the value of thickness needed to evaluate V_i^{n+1} and I_{ij}^{n+1}, B_i^{proj} is the estimated value of B_i^{n+1}, and $BASE_i$ is the value of BASE(x,y) at node i.

Now the spatially integrated equation, (3.28), may be written discretely in time:

$$S_i V_i^{n+1} \left(\frac{h_i^{n+1} - h_i^n}{\Delta t_{n+1}} \right) + \sum_{j=1}^{NN} I_{ij}^{n+1} h_j^{n+1} = Q_{IN_i} + Q_i^* \qquad (3.35)$$

$$i = \overline{1,NN}$$

where V_i^{n+1} and I_{ij}^{n+1} are evaluated based on projected thickness, B_i^{proj}.

3.5 Boundary Conditions and Solution of Discretized Equation

Matrix equation and solution sequence

The NN relations given by (3.35) may be rearranged and rewritten in matrix form:

$$\left(\frac{1}{\Delta t_{n+1}}\right)\begin{bmatrix} S_1V_1^{n+1} & 0 & 0 & \cdots & 0 \\ 0 & S_2V_2^{n+1} & 0 & \cdots & 0 \\ 0 & 0 & S_3V_3^{n+1} & & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & S_{NN}V_{NN}^{n+1} \end{bmatrix}\begin{Bmatrix} h_1^{n+1} \\ h_2^{n+1} \\ h_3^{n+1} \\ \vdots \\ h_{NN}^{n+1} \end{Bmatrix}$$

$$+\begin{bmatrix} I_{11}^{n+1} & I_{12}^{n+1} & I_{13}^{n+1} & I_{14}^{n+1} & \cdot & \cdot & \cdot & \cdot & I_{1,NN}^{n+1} \\ I_{21}^{n+1} & I_{22}^{n+1} & I_{23}^{n+1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ I_{31}^{n+1} & I_{32}^{n+1} & I_{33}^{n+1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ I_{41}^{n+1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ I_{NN,1}^{n+1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & I_{NN,NN}^{n+1} \end{bmatrix}\begin{Bmatrix} h_1^{n+1} \\ h_2^{n+1} \\ h_3^{n+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ h_{NN}^{n+1} \end{Bmatrix}$$

$$=\left(\frac{1}{\Delta t_{n+1}}\right)\begin{Bmatrix} S_1 & V_1^{n+1} & h_1^n \\ S_2 & V_2^{n+1} & h_2^n \\ S_3 & V_3^{n+1} & h_3^n \\ \vdots & & \\ S_{NN} & V_{NN}^{n+1} & h_{NN}^n \end{Bmatrix}+\begin{Bmatrix} Q_{IN_1} \\ Q_{IN_2} \\ Q_{IN_3} \\ \vdots \\ Q_{IN_{NN}} \end{Bmatrix}+\begin{Bmatrix} Q_1^* \\ Q_2^* \\ Q_3^* \\ \vdots \\ Q_{NN}^* \end{Bmatrix} \qquad (3.36)$$

By adding the two matrices on the left side, and the vectors on the right side, a matrix equation is obtained which may be solved for the model heads at the new time level, t^{n+1}, on each time step:

$$
\begin{bmatrix}
\left(\frac{S_1 V_1^{n+1}}{\Delta t_{n+1}} + I_{11}^{n+1}\right) & I_{12}^{n+1} & I_{13}^{n+1} & \cdots & I_{1,NN}^{n+1} \\
I_{21}^{n+1} & \left(\frac{S_2 V_2^{n+1}}{\Delta t_{n+1}} + I_{22}^{n+1}\right) & I_{23}^{n+1} & \cdots & \vdots \\
I_{31}^{n+1} & I_{32}^{n+1} & \ddots & & \vdots \\
\vdots & \vdots & & & \vdots \\
I_{NN,1}^{n+1} & \cdots & & & \left(\frac{S_{NN} V_{NN}^{n+1}}{\Delta t_{n+1}} + I_{NN,NN}^{n+1}\right)
\end{bmatrix}
\begin{Bmatrix}
h_1^{n+1} \\ h_2^{n+1} \\ h_3^{n+1} \\ \vdots \\ h_{NN}^{n+1}
\end{Bmatrix}
$$

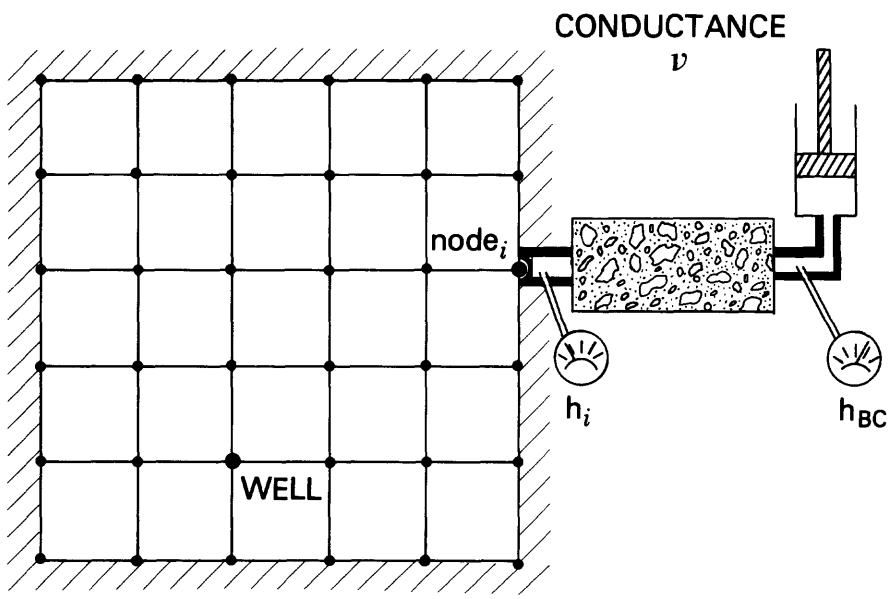
$$
= \begin{Bmatrix}
\frac{S_1 V_1^{n+1} h_1^n}{\Delta t_{n+1}} + Q_{IN_1} + Q_1^* \\
\frac{S_2 V_2^{n+1} h_2^n}{\Delta t_{n+1}} + Q_{IN_2} + Q_2^* \\
\vdots \\
\frac{S_{NN} V_{NN}^{n+1} h_{NN}^n}{\Delta t_{n+1}} + Q_{IN_{NN}} + Q_{NN}^*
\end{Bmatrix} \qquad (3.37)
$$

The solution progresses through time as follows: On a given time step, the nodal heads at the beginning of the step are known values and are placed in h_j^n on the right hand side vector of (3.37). The thickness-dependent values are determined based on the projection of B in (3.34) using projected head of (3.33). The integrals and volumes are evaluated and the matrix and vector completed. The nodal heads at the end of the current time step are solved for by Gaussian elimination for the (banded) matrix on the left of (3.37). The new heads are then placed on the right side of (3.37) into h^n, and a new time step is begun.

Specification of boundary conditions

Before solving the matrix equation as described above, information about boundary conditions must be included. In the case of solving for heads, the boundary conditions take the form of either specified fluid fluxes across boundaries which are directly entered in the terms, Q_{IN_i}, or of particular head values specified at nodal locations. At a point of fixed head in an aquifer, a particular value of fluid inflow or outflow occurs at that point in order to keep the head constant when the aquifer is stressed. It is this flux of fluid which is added to the model aquifer in order to obtain fixed heads at nodes.

Consider the closed system of Figure 3.5 in which head at node i, h_i, is to have a specified value, h_{BC}, for all time. A well is removing water from the system at an internal node. A core of porous medium with conductance ν is connected to node i. The head outside the core is held at the specified value, h_{BC}. The head at node i, h_i, is calculated by the model. A flow of Q_{BC_i} $[L^3/s]$ enters through the core at node i in order to balance the rate of fluid removal at the well. The resulting head at node i depends on the conductance value ν of the core. If ν is very small, then a large head drop is required across the core in order to supply fluid at the rate the pumping well requires. This results in h_i having quite a different value from h_{BC}. If, however, ν is very large, then the value of head at node i, is very close to h_{BC}, as only a minute head drop across the core supplies the fluid required by the well. Therefore, by applying flux to a node through a highly conductive core, the outside of which is held at a specified head value, the node responds with a head value nearly equal to that specified. An advantage of specifying head this way



$$\text{INFLOW} = Q_{BC_i} = v\,(h_{BC} - h_i)$$

Figure 3.5
Schematic representation of specified head (or pressure) boundary condition.

is that when head at a node in the mesh is fixed, a calculation of the flux entering the mesh at this node is obtained at the same time.

This flux is defined as follows:

$$Q_{BC_i} = \nu \left(h_{BC_i} - h_i^{n+1} \right) \tag{3.38}$$

where Q_{BC_i} is the inflow at node i resulting from the specified head boundary condition, ν is the conductance of the 'core', and h_{BC_i} is the specified value of head at node i on the boundary.

The matrix equation (3.37) may be written in short form as:

$$\sum_{j=1}^{NN} M_{ij}^{n+1} h_j^{n+1} = \left(\frac{S_i V_i^{n+1}}{\Delta t_{n+1}} \right) h_i^n + Q_i^* + Q_{IN_i} + Q_{BC_i} \tag{3.39}$$

$$i = \overline{1,NN}$$

wherein an additional flux Q_{BC_i} has been added to account for specified head nodes. At such a node, say node A, the equation is:

$$\sum_{j=1}^{NN} M_{Aj}^{n+1} h_j^{n+1} = \left(\frac{S_A V_A^{n+1}}{\Delta t_{n+1}} \right) h_A^n + Q_A^* + Q_{IN_A} + \nu \left(h_{BC_A} - h_A^{n+1} \right) \tag{3.40}$$

where ν is very large, then the last term dominates the equation and (3.40) becomes:

$$h_A^{n+1} \simeq h_{BC_A} \tag{3.41}$$

Thus the specified head is set at node A, but as h_A^{n+1} and h_{BC_A} are slightly different, a flux may be determined from (3.38).

DETAILS OF

SUTRA

METHODOLOGY

(p.95 follows)

Chapter 4

Numerical Methods

In this section, the numerical methods upon which SUTRA is based are presented in detail. The purpose of this presentation is to provide a complete reference for the computer code.

4.1 Basis and Weighting Functions

Basis functions, weighting functions and their derivatives are all described in local element geometry. In a local coordinate system, every element takes the shape of a two by two square. The local coordinates, ξ and η, are shown along with a generic local finite element in Figure 4.1. The origin of the local coordinate system is at the center of the element. Local node one always has local coordinates $(\xi, \eta) = (-1, -1)$. The other nodes are numbered counter-clockwise from the first node as shown in Figure 4.1.

The following one-dimensional basis functions are defined over the region of the element:

$$\Xi_-(\xi) = \frac{1}{2}(1 - \xi) \tag{4.1}$$

$$\Xi_+(\xi) = \frac{1}{2}(1 + \xi) \tag{4.2}$$

$$H_-(\eta) = \frac{1}{2}(1 - \eta) \tag{4.3}$$

$$H_+(\eta) = \frac{1}{2}(1 + \eta) \tag{4.4}$$

These linear one-dimensional basis functions are continuous in ξ and η and have either a value of zero or one depending on whether ξ or η have a value of +1 or -1. The one-dimensional functions are combined to create the bi-linear basis functions used in SUTRA:

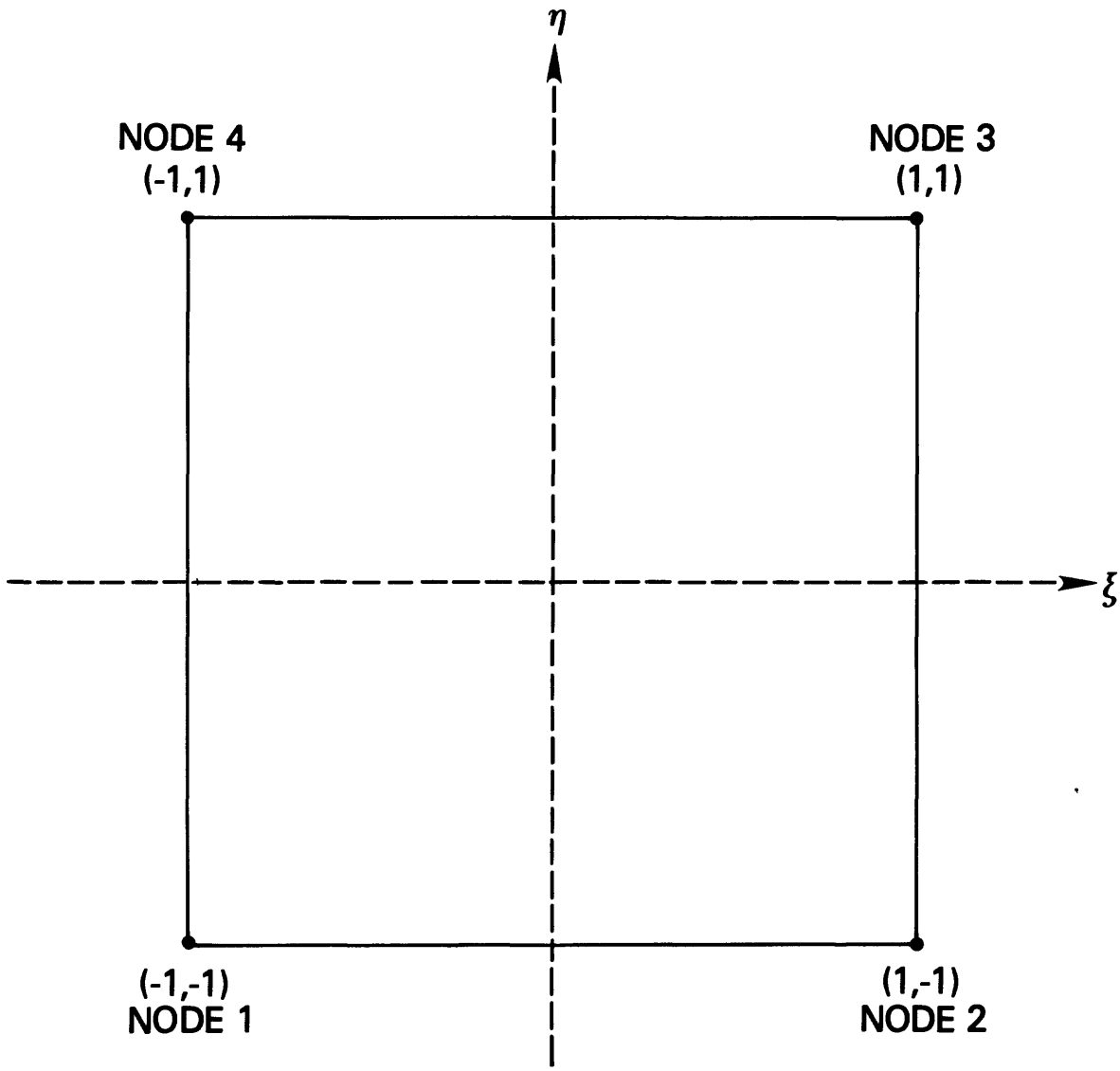


Figure 4.1
Quadrilateral finite element in local coordinate system (ξ,η).

$$\Omega_1(\xi,\eta) = \Xi_- H_- \tag{4.5}$$

$$\Omega_2(\xi,\eta) = \Xi_+ H_- \tag{4.6}$$

$$\Omega_3(\xi,\eta) = \Xi_+ H_+ \tag{4.7}$$

$$\Omega_4(\xi,\eta) = \Xi_- H_+ \tag{4.8}$$

The two-dimensional bi-linear basis functions, when defined in the local element coordinate system are denoted as $\Omega_i(\xi,\eta)$, i=1,2,3,4. There is one basis function defined for each node.

The basis function Ω_i, defined for node i, has a value of one at the node and a value of zero at the other nodes. The surface representing $\Omega_i(\xi,\eta)$ over an element is curved due to the product of ξ and η in equations (4.5) through (4.8). A trajectory in the surface parallel to an element side, however, is a perfectly straight line as shown in Figure 4.2. This is born out in the derivatives of the bi-linear basis functions which depend on only one space coordinate:

$$\frac{\partial\Omega}{\partial\xi}1 = -\frac{1}{2} H_- \qquad \frac{\partial\Omega}{\partial\eta}1 = -\frac{1}{2} \Xi_- \tag{4.9}$$

$$\frac{\partial\Omega}{\partial\xi}2 = +\frac{1}{2} H_- \qquad \frac{\partial\Omega}{\partial\eta}2 = -\frac{1}{2} \Xi_+ \tag{4.10}$$

$$\frac{\partial\Omega}{\partial\xi}3 = +\frac{1}{2} H_+ \qquad \frac{\partial\Omega}{\partial\eta}3 = +\frac{1}{2} \Xi_+ \tag{4.11}$$

$$\frac{\partial\Omega}{\partial\xi}4 = -\frac{1}{2} H_+ \qquad \frac{\partial\Omega}{\partial\eta}4 = +\frac{1}{2} \Xi_- \tag{4.12}$$

Asymmetric weighting functions are defined for use in a Galerkin-Petrov method (one version of which is described in Huyakorn and Pinder, 1983). These are not applied for nodewise discretization of parameters, but rather for weighting in the volume integrals of the governing equation. They may be used

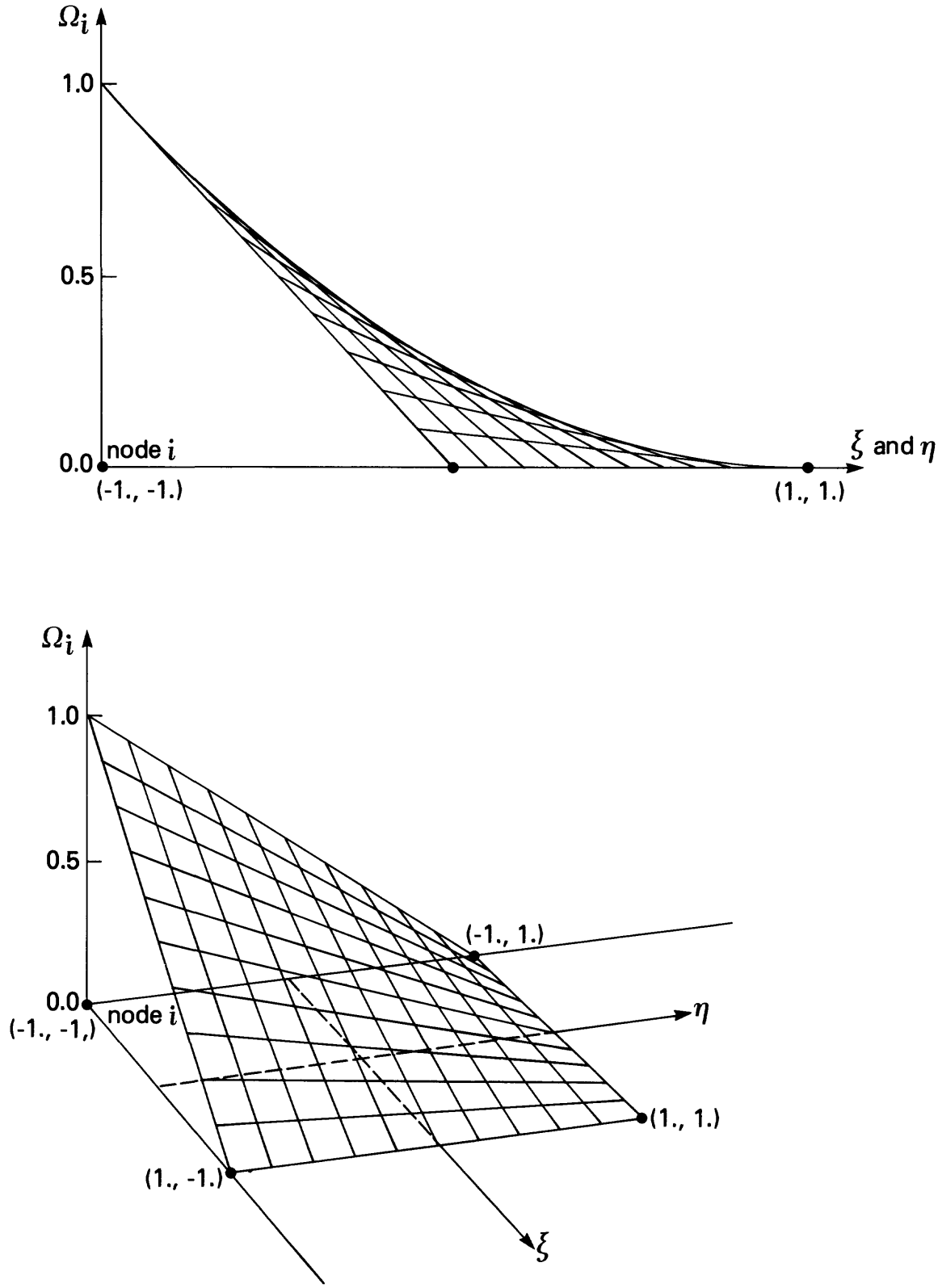


Figure 4.2
Perspectives of basis function $\Omega_i(\xi,\eta)$ at node i.

to give an 'upstream weighting' to the advective flux term in the transport equations or to provide 'upstream weighting' to the fluid flux term in the fluid mass balance when the medium is unsaturated. The asymmetric functions are defined as follows:

$$\theta_1(\xi,\eta) = \left(\Xi_- - \Xi^*\right)\left(H_- - H^*\right) \tag{4.13}$$

$$\theta_2(\xi,\eta) = \left(\Xi_+ + \Xi^*\right)\left(H_- - H^*\right) \tag{4.14}$$

$$\theta_3(\xi,\eta) = \left(\Xi_+ + \Xi^*\right)\left(H_+ + H^*\right) \tag{4.15}$$

$$\theta_4(\xi,\eta) = \left(\Xi_- - \Xi^*\right)\left(H_+ + H^*\right) \tag{4.16}$$

where:

$$\Xi^* = 3a_\xi\, \Xi_-\, \Xi_+ \tag{4.17}$$

$$H^* = 3a_\eta\, \Xi_-\, \Xi_+ \tag{4.18}$$

The spatial derivatives are:

$$\frac{\partial\theta}{\partial\xi}1 = -\frac{1}{2}\left(1-3a_\xi\xi\right)\left(H_- - H^*\right) \qquad \frac{\partial\theta}{\partial\eta}1 = -\frac{1}{2}\left(1-3a_\eta\eta\right)\left(\Xi_- - \Xi^*\right) \tag{4.19}$$

$$\frac{\partial\theta}{\partial\epsilon}2 = +\frac{1}{2}\left(1-3a_\xi\xi\right)\left(H_- - H^*\right) \qquad \frac{\partial\theta}{\partial\eta}2 = -\frac{1}{2}\left(1-3a_\eta\eta\right)\left(\Xi_+ + \Xi^*\right) \tag{4.20}$$

$$\frac{\partial\theta}{\partial\xi}3 = +\frac{1}{2}\left(1-3a_\xi\xi\right)\left(H_+ + H^*\right) \qquad \frac{\partial\theta}{\partial\eta}3 = +\frac{1}{2}\left(1-3a_\eta\eta\right)\left(\Xi_+ + \Xi^*\right) \tag{4.21}$$

$$\frac{\partial\theta}{\partial\xi}4 = -\frac{1}{2}\left(1-3a_\xi\xi\right)\left(H_+ + H^*\right) \qquad \frac{\partial\theta}{\partial\eta}4 = +\frac{1}{2}\left(1-3a_\eta\eta\right)\left(\Xi_- - \Xi^*\right) \tag{4.22}$$

The parameters a_ξ and a_η determine the amount of asymmetry (or upstream weight) in each coordinate direction. When these parameters have a value of zero, then the basis functions and their derivatives, equivalent to (4.5) through (4.12) are exactly obtained from (4.13) through (4.22). The values of a_ξ and a_η depend on location in the element:

$$a_\xi(\xi,\eta) = (UP)\left(\frac{v_\xi}{|\underline{v}_{local}|}\right) \tag{4.23}$$

$$a_\eta(\xi,\eta) = (UP)\left(\frac{v_\eta}{|\underline{v}_{local}|}\right) \tag{4.24}$$

where UP is the fractional strength of upstream weighting desired (chosen by the model user), $v_\xi(\xi,\eta)$ and $v_\eta(\xi,\eta)$ are the components of fluid velocity given in terms of local element coordinates, and $|\underline{v}_{local}(\xi,\eta)|$ is the magnitude of fluid velocity given in terms of local coordinates. Each velocity component may vary in value throughout the element. A description of the calculation of fluid velocity is given in section 4.6, "Consistent Evaluation of Fluid Velocity."

Note that the basis functions, weighting functions and their derivatives are calculated by the SUTRA subroutine 'BASIS2'.

4.2 Coordinate Transformations

During calculations for the finite-element mesh and during integral evaluations, transformations are required between the global (x,y) coordinate system in which an element may have an arbitrary size and quadrilateral shape, and the local (ξ,η) coordinate system in which each element is a two by two square. Transformations are required in both directions. The transformation

involves a linear remapping in each coordinate direction and employs the basis functions to provide mapping. The Jacobian matrix [J] is calculated separately for each element that requires transformation and may vary from point to point in an element.

$$\left[J \right] = \begin{bmatrix} \frac{\partial \Omega}{\partial \xi}_1 & \frac{\partial \Omega}{\partial \xi}_2 & \frac{\partial \Omega}{\partial \xi}_3 & \frac{\partial \Omega}{\partial \xi}_4 \\ \frac{\partial \Omega}{\partial \eta}_1 & \frac{\partial \Omega}{\partial \eta}_2 & \frac{\partial \Omega}{\partial \eta}_3 & \frac{\partial \Omega}{\partial \eta}_4 \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix} \tag{4.25}$$

The numbered subscripts refer to the local element numbering of Figure 4.1.

The Jacobian matrix is used to transform derivatives of basis functions from the global to the local coordinate systems and the reverse:

$$\begin{Bmatrix} \frac{\partial \Omega}{\partial \xi}_j \\ \frac{\partial \Omega}{\partial \eta}_j \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial \phi}{\partial x}_j \\ \frac{\partial \phi}{\partial y}_j \end{Bmatrix} \tag{4.26}$$

$$\begin{Bmatrix} \frac{\partial \phi}{\partial x}_j \\ \frac{\partial \phi}{\partial y}_j \end{Bmatrix} = \left[J^{-1} \right] \begin{Bmatrix} \frac{\partial \Omega}{\partial \xi}_j \\ \frac{\partial \Omega}{\partial \eta}_j \end{Bmatrix} \tag{4.27}$$

where:

$$\left[J \right] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \tag{4.28}$$

The subscript j refers to any one of the four nodes in an element and ϕ_j refers to the global basis function as defined for the j^{th} node in an element. The same transformations apply to derivatives of the asymmetric weighting functions which are denoted w_j in global coordinates. In (4.27), $\left[J^{-1} \right]$ is the inverse Jacobian matrix defined as:

$$\left[J^{-1} \right] = \left(\frac{1}{\det J} \right) \begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix} \tag{4.29}$$

where det J is the determinant of the Jacobian given by:

$$\det J = J_{11} J_{22} - J_{12} J_{21} \tag{4.30}$$

The determinant may vary bi-linearly over an element.

Differential elements of area, dA, are transformed between local and global coordinate systems as:

$$dA = dx\, dy = (\det J)\, d\xi\, d\eta \tag{4.31}$$

Note that the Jacobian matrix, determinant of the Jacobian, and the derivatives of the basis functions in local and global coordinates are calculated in SUTRA subroutine, 'BASIS2'.

4.3 Gaussian Integration

Gaussian integration is a method by which exact integration of polynomials may be carried out through a simple summation of point values of the integrand. The method is:

$$\int_{\tau=-1}^{\tau=+1} f(\tau)\, d\tau = \sum_{KG=1}^{NP} G_{KG}\, f(\tau_{KG}) \tag{4.32}$$

where $f(\tau)$ is the function to be integrated between $\tau = -1$ and $\tau = +1$. KG is the Gauss point number, NP is the total number of Gauss points, G_{KG} is a constant, and τ_{KG} is the location of the KG^{th} Gauss point. An exact integration is guaranteed by the sum in (4.32) if (2n-1) Gauss points are used for a polynomial $f(\tau)$ of order n. For evaluation of integrals which arise in the SUTRA methodology, only two Gauss points are used in a given coordinate direction as the integrals

encountered are usually of order three or less. In this case, the constants, G_{KG} have a value of one and (4.32) simplifies to:

$$\int_{\tau=-1}^{\tau=+1} f(\tau)\, d\tau = \sum_{KG=1}^{2} f(\tau_{KG}) \tag{4.33}$$

The values of τ_{KG} for Gauss points one and two, are minus and plus 0.577350269189626, (or $\pm 3^{-\frac{1}{2}}$ respectively).

The need to define a two by two element in local coordinates is apparent here. Gaussian integration is done over a range of two from -1 to +1. In order to integrate a term of the differential governing equation over an arbitrary quadrilateral element in the mesh, the limits of the integral must first be transformed to values of -1 and +1, that is, to local coordinates. When integrating a double integral, both integrals must be transformed to have limits of -1 and +1, and two Gauss points are needed in each coordinate direction. These are defined as shown in Figure 4.3.

An example, evaluating the integral of (3.24) follows: The integral to evaluate is:

$$A_{ij} = \int_{x_L}\int_{y_L} (\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i)\, B_i\, dy\, dx \tag{4.34}$$

where x_L and y_L indicate that the integral is over the area of an element L in global coordinates. First, the (x,y) integral is converted to an integral in local coordinates (ξ,η) through use of the Jacobian:

$$A_{ij} = \int_{\xi=-1}^{+1}\int_{\eta=-1}^{+1} (\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i)\, B_i\, (\det J)\, d\eta\, d\xi \tag{4.35}$$

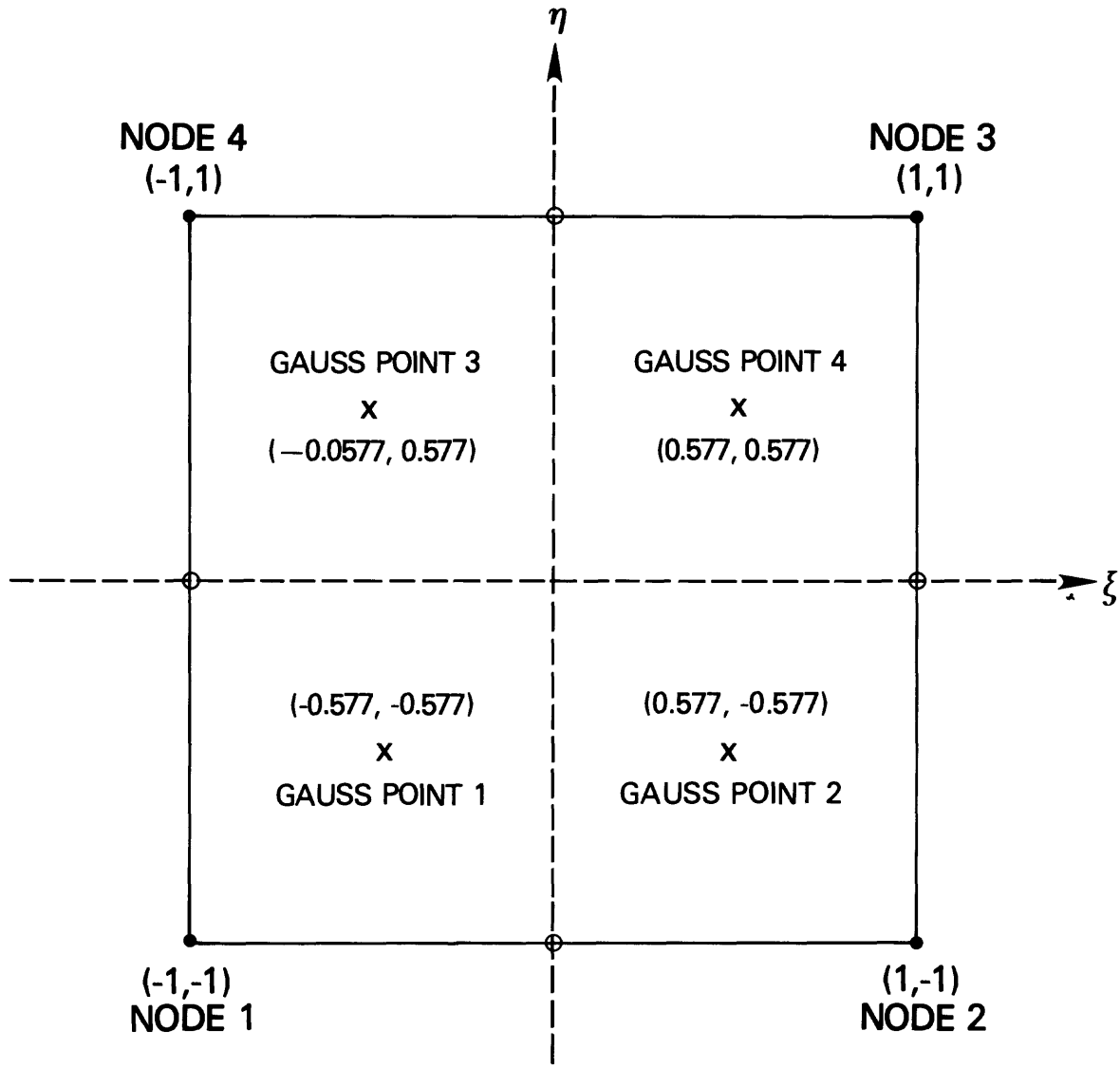


Figure 4.3
Finite element in local coordinate system with Gauss points.

The values of $\underline{\nabla}\phi$ are in global coordinates and are obtained by transformation of derivatives calculated in local coordinates.

Gaussian integration is applied independently to each integral:

$$A_{ij} = \sum_{K_\xi=1}^{2} \sum_{K_\eta=1}^{2} \left[\left(\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i \right) B_i \, (\det J) \right]_{\left(\xi_{K_\xi}, \eta_{K_\eta} \right)} \qquad (4.36)$$

or equivalently as a single summation:

$$A_{ij} = \sum_{KG=1}^{4} \left[\left(\underline{\nabla}\phi_j \cdot \underline{\nabla}\phi_i \right) B_i \, (\det J) \right]_{\left(\xi_{KG}, \eta_{KG} \right)} \qquad (4.37)$$

where K_ξ and K_η refer to Gauss point locations in the ξ and η directions, and where ξ_{KG} and η_{KG} refer to the four Gauss points arising in (4.36) as depicted in Figure 4.3. Thus, in order to evaluate the integral (4.34) over a given element, only four values of the integral need to be summed as given in (4.37), with one value determined at each of the four Gauss points.

In the case where an element is a non-rectangular quadrilateral with variable thickness B, the polynomial to be integrated in (4.35) is of fourth order as each of the terms may vary linearly in the same direction. Otherwise it is always of third order or less, and two-point Gauss integration provides exact results.

Note that the summation indication by (4.37) over the Gauss points is carried out by SUTRA subroutine 'ELEMEN' for each element in the mesh and for each integral which requires evaluation.

4.4 Numerical Approximation of SUTRA Fluid Mass Balance

The governing equation representing the SUTRA fluid mass balance (2.24), is modified by the addition of a point source term which is used to insert points at which pressure is specified. This is done as described in text referring to relation (3.38).

$$O_p(p,U) = \left(S_w \rho S_{op} + \varepsilon\rho \frac{\partial S}{\partial p_w}\right)\frac{\partial p}{\partial t} + \left(\varepsilon S_w \frac{\partial \rho}{\partial U}\right)\frac{\partial U}{\partial t}$$

$$- \underline{\nabla} \cdot \left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right) \cdot (\underline{\nabla} p - \rho \underline{g}) \right] - Q_p \qquad (4.38)$$

$$- \nu_p \left(p_{BC} - p\right) = 0$$

The last term is the source term arising from a specified pressure condition, wherein ν_p is a 'conductance' and $p_{BC}(t)$ is the externally specified pressure boundary condition value. When ν_p is set to a sufficiently large value, the last term becomes much larger than the others in (4.38), and $p \approx p_{BC}$, which is the desired boundary condition. Relation (4.38) is numerically approximated in the following sections.

Spatial integration

When the equation for $O_p(p,U)$ is approximated through nodewise, elementwise and cellwise discretizations, it no longer exactly equals zero. The approximate equation, $\hat{O}_p(p,U)$, equals a spatially varying residual, $R_p(x,y,t)$, as shown in (3.8). A weighted residual formulation may be written as:

$$\int_V \hat{\hat{O}}_p(p,U)\, W_i(x,y)\, dV = 0 \qquad i = \overline{1,NN} \tag{4.39}$$

where $W_i(x,y)$ is the weighting function in global coordinates chosen to be either the basis function, $\phi_i(x,y)$ or the asymmetric weighting function, $\omega_i(x,y)$, depending on the term of the equation. Relation (4.38) is approximated discretely and substituted for $\hat{\hat{O}}_p(p,U)$ in (4.39). The resulting set of integral terms is evaluated, one term at a time in the following paragraphs.

The first term is an integral of the pressure derivative:

$$\int_V \left[\left(S_w \rho S_{op} + \varepsilon \rho \frac{\partial S}{\partial p} w \right) \frac{\partial p}{\partial t} \right] \phi_i(x,y)\, dV \tag{4.40}$$

where the term in brackets is discretized cellwise, with one value of the term for each of the NN cells in the mesh, and the weighting function is chosen to be the basis function (written in global coordinates) The carat (^) or large carat ($\hat{\hat{}}$) over a term indicates that it has been approximated in one of the three ways. Because the cellwise-approximated term is constant for a node i, it is removed from the integral leaving only the basis function to be integrated. The volume integral of $\phi_i(x,y)$ gives the volume V_i of cell i according to relation (3.15). The term (4.40) becomes:

$$\left(S_w \rho S_{op} + \varepsilon \rho \frac{\partial S}{\partial p} w \right)_i \frac{\partial p}{\partial t}_i V_i \tag{4.41}$$

The second term of the expanded form of (4.39) is also a time derivative which is approximated cellwise:

$$\int_V \left[\left(\varepsilon S_w \hat{\hat{\frac{\partial \rho}{\partial U}}} \right) \frac{\partial U}{\partial t} \right] \phi_i(x,y)\, dV = \left(\varepsilon S_w \frac{\partial \rho}{\partial U} \right)_i \frac{\partial U}{\partial t}_i V_i \tag{4.42}$$

The third term of expanded relation (4.39) involving the divergence of fluid flux is weighted with the asymmetric function. The asymmetry is intended for use only in unsaturated flow problems to maintain solution stability when the mesh has not been designed fine enough to represent sharp saturation fronts. In general, the usual symmetric function is used for weighting this flux term even for unsaturated flow, but the term is developed with the asymmetric function in order to provide generality. Green's Theorem (3.17) is applied yielding:

$$
\begin{aligned}
&-\int_V \left\{ \underline{\nabla}\cdot\left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right)\hat{} \cdot \left(\underline{\nabla}p - \rho\underline{g}\right)\right]\right\} \omega_i(x,y)\, dV \\
&= -\int_\Gamma \left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right)\hat{} \cdot \left(\underline{\nabla}p - \rho\underline{g}\right)\right] \cdot \underline{n}\, \omega_i(x,y)\, d\Gamma \\
&+\int_V \left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right)\hat{} \cdot \left(\underline{\nabla}p - \rho\underline{g}\right)\right] \cdot \underline{\nabla}\omega_i\, dV
\end{aligned}
\qquad (4.43)
$$

wherein the terms with carats are approximated discretely as described below, $\underline{n}$ is the unit outward normal to the three-dimensional surface bounding the region to be simulated, and Γ is the surface of the region. The asymmetric weighting function in global (rather than local) coordinates is denoted, $\omega_i(x,y)$. The first term on the right of (4.43) is exactly the fluid mass flux (see Darcy's law, relation (2.19)) out across the region's boundary at node i, $q_{OUT_i}(t)$ in units of [M/s]:

$$
q_{OUT_i}(t) = \int_\Gamma \left(\varepsilon\rho\hat{\underline{v}}\cdot\underline{n}\right)\omega_i\, d\Gamma = \int_\Gamma -\left[\left(\frac{\underline{\underline{k}}\, k_r \rho}{\mu}\right)\hat{} \cdot \left(\underline{\nabla}p - \rho\underline{g}\right)\right] \cdot \underline{n}\, \omega_i\, d\Gamma \qquad (4.44)
$$

This term is used to specify fluid flows across boundaries in SUTRA. Note that an inflow, $q_{IN_i}(t)$ is $q_{IN_i} = -q_{OUT_i}$.

The second term on the right of (4.43) is approximated using a combination of elementwise and nodewise discretizations. The approximation of $(\underline{\nabla}p - \rho\underline{g})$ requires particular attention and is discussed in section 4.6, "Consistent Evaluation of Fluid Velocity." The permeability tensor appearing in (4.43) in general has nine components, however, $(\underline{\nabla}p - \rho\underline{g})$ is always zero in the third spatial direction due to the assumption of a two-dimensional model. Thus only four components of the permeability tensor are required:

$$\hat{\underline{\underline{k}}}^L = \begin{bmatrix} k^L_{xx} & k^L_{xy} \\ k^L_{yx} & k^L_{yy} \end{bmatrix} \tag{4.45}$$

wherein $\underline{\underline{k}}$ and is discretized elementwise as indicated by $\hat{\underline{\underline{k}}}^L$. The pressure is discretized nodewise:

$$p(x,y,t) \simeq \sum_{i=1}^{NN} p_i(t)\, \phi_i(x,y) \tag{4.46}$$

Relative permeability, k_r, depends on saturation which, in turn, depends on pressure. Relative permeabilities are evaluated at each Gauss point during numerical integration depending on the saturation (and pressure) at the Gauss point. Viscosity is evaluated at each Gauss point for energy transport as a function of nodewise discretized temperature, and is constant for solute transport.

Density, ρ, when it appears in the permeability term, is also evaluated at each Gauss point depending on the nodewise discretized value of U at the Gauss point. The density appearing in product with the gravity term is expressly not evaluated in this usual manner. A particular discretization is used which maintains consistency with the $\underline{\nabla}p$ term as described in section 4.6, "Consistent Evaluation of Fluid Velocity". This consistently-evaluated $\rho\underline{g}$ term is denoted $\hat{\rho\underline{g}}^*$, (see relation (4.103)).

The second term on the right of (4.43) is thus approximated as:

$$\sum_{j=1}^{NN} p_j(t) \int_x \int_y \left\{\left[\left(\hat{\underline{\underline{k}}}^L\right)\left(\frac{k_r\rho}{\mu}\right)\right]\cdot\underline{\nabla}\phi_j\right\}\cdot\underline{\nabla}\omega_i \; B(x,y)\, dy\, dx \qquad (4.47)$$

$$-\int_x \int_y \left\{\left[\left(\hat{\underline{\underline{k}}}^L\right)\left(\frac{k_r\rho}{\mu}\right)\right]\cdot\left[\hat{(\rho\underline{g})}^*\right]\cdot\underline{\nabla}\omega_i \; B(x,y)\, dy\, dx$$

where $\hat{\underline{\underline{k}}}^L$ indicates an elementwise discretized permeability tensor, $\left(\frac{k_r\rho}{\mu}\right)$ indicates the value of the term based on nodewise discretized values of p and U, and $\hat{(\rho\underline{g})}^*$ indicates a discretization of $(\rho\underline{g})$ consistent with the discretization of $\underline{\nabla}p$. The thickness of the mesh, B(x,y), is evaluated at each Gauss point depending on a nodewise discretization:

$$B(x,y) \simeq \sum_{i=1}^{NN} B_i \, \phi_i(x,y) \qquad (4.48)$$

where B_i is the mesh thickness at node i. Note that mesh thickness is fixed and may not vary in time as was allowed for illustrative purposes in Chapter 3, "Fundamentals of Numerical Algorithms."

The last two terms of (4.38) are approximated cellwise with a basis function for weighting.

$$- \int_V \hat{Q}_p \, \phi_i(x,y) \, dV - \int \left[\nu_p \left(\hat{p}_{BC} - p \right) \right] \phi_i(x,y) \, dV = - Q_i - \nu_i \left(p_{BC_i} - p_i \right) \tag{4.49}$$

The cellwise discretizations which are employed in the above evaluations are:

$$\hat{Q}_p = \sum_{i=1}^{NN} \left(\frac{Q_i}{V_i} \right) \tag{4.50}$$

$$\left[\hat{Q}_{PBC} \right] = \left[\nu_p \left(\hat{p}_{BC} - p \right) \right] = \sum_{i=1}^{NN} \left[\left(\frac{\nu_i}{V_i} \right) \left(p_{BC_i} - p_i \right) \right] \tag{4.51}$$

where V_i is the volume of cell i, $Q_i(t)$ [M/s] is the total mass source to cell i, Q_{PBC} [$M/L^3 \cdot s$] is the fluid mass source rate due to the specified pressure, and ν_i[L·s] is the pressure-based conductance for the specified pressure source in cell i. The conductance is set to zero for nodes at which pressure is not specified, and to a high value at nodes where pressure is specified.

By combining and rearranging the evaluations of approximate terms of (4.39), the following weighted residual relation is obtained:

$$AF_i \frac{dp}{dt}i + CF_i \frac{dU}{dt}i + \sum_{j=1}^{NN} p_j(t) \, BF_{ij} + \nu_i p_i = Q_i + \nu_i p_{BC_i} + q_{IN_i} + DF_i$$

$$i = \overline{1,NN} \tag{4.52}$$

where:

$$AF_i = \left(S_w \rho S_{op} + \varepsilon \rho \frac{\partial S}{\partial p} w \right)_i V_i \tag{4.53}$$

$$CF_i = \left(\varepsilon S_w \frac{\partial \rho}{\partial U} \right)_i V_i \tag{4.54}$$

$$BF_{ij} = \int_x \int_y \left\{ \left[\left(\hat{\underline{\underline{k}}}^L \right) \left(\frac{\hat{k}_r \rho}{\mu} \right) \right] \cdot \underline{\nabla} \phi_j \right\} \cdot \underline{\nabla} \omega_i \, B \, dy \, dx \tag{4.55}$$

$$DF_i = \int_x \int_y \left\{ \left[\left(\hat{\underline{\underline{k}}}^L \right) \left(\frac{\hat{k}_r \rho}{\mu} \right) \right] \cdot \left[\left(\hat{\rho \underline{g}}^* \right) \right] \right\} \cdot \underline{\nabla} \omega_i \, B \, dy \, dx \tag{4.56}$$

The only integrals requiring Gaussian integration are BF_{ij} and DF_i. Note that these are evaluated in SUTRA subroutine ELEMEN in an element by element manner. The other terms except for those involving ν_i are evaluated cellwise (one for each node). Note that this is done by subroutine NODALB, and the specified pressure terms are evaluated by subroutine BCB.

Temporal discretization and iteration

The time derivatives in the spatially discretized and integrated equation are approximated by finite differences. The pressure term is approximated as:

$$\frac{dp_i}{dt} \simeq \frac{p_i^{n+1} - p_i^n}{\Delta t_{n+1}} \tag{4.57}$$

where

$$p_i^n = p_i(t^n) \tag{4.58a}$$

$$p_i^{n+1} = p_i(t^n + \Delta t_{n+1}) = p_i(t^{n+1}) \tag{4.58b}$$

and

$$\Delta t_{n+1} = t^{n+1} - t^n \tag{4.59}$$

The new or current time step, Δt_{n+1}, begins at time t^n and ends at time t^{n+1}. The previous time step for which a solution has already been obtained at time t^n is denoted Δt_n.

The term in (4.52) involving the time derivative of concentration or temperature, $\frac{dU}{dt}$, makes only a very small contribution to the fluid mass balance. For solution over the present time step, Δt_{n+1}, this derivative is evaluated using information from the previous time step, as these values are already known:

$$\frac{dU}{dt}_i \approx \left(\frac{dU}{dt}_i\right)^n = \frac{U_i^n - U_i^{n-1}}{\Delta t_n} \tag{4.60}$$

This approximation gives a simple method of accounting for this small contribution to the fluid mass balance.

All other terms in (4.51) are evaluated at the new time level t^{n+1} for solution of the present time step, Δt_{n+1}, except for the density in the consistently discretized $\left(\hat{\rho}^* \underline{g}\right)$ term. The density is evaluated based on $U(t^n)$, the value of U at the beginning of the present time step. Because coefficients depend on the, as yet, unknown values of p and U at the end of the time step, one or more iterations may be used to solve this non-linear problem. On the first iteration, and when only one iteration per time step is used, coefficents are based on a projected value of p and U.

$$p_i^{proj} = p_i^n + \left(\frac{\Delta t_{n+1}}{\Delta t_n}\right)\left(p_i^n - p_i^{n-1}\right) \tag{4.61}$$

$$U_i^{proj} = U_i^n + \left(\frac{\Delta t_{n+1}}{\Delta t_n}\right)\left(U_i^n - U_i^{n-1}\right) \tag{4.62}$$

These projections estimate the p and U values at a node i, p_i^{proj} and U_i^{proj}, at the end of the present time step, Δt_{n+1}, based on linear extrapolation of the two previous values of p and U. All p and U dependent coefficients (except $\hat{\rho}^* \underline{g}$) in (4.52) through (4.56) are estimated at time level t^{n+1}. These coefficient values are based on the most recent values of p and U, be they projections or solutions to the previous iteration. Iterations end when the maximum change in p and U at any node in the mesh falls below user-specified criteria of absolute change in p and U.

The weighted residual relations (4.52) may thus be written in a form which allows for solution of pressures at nodes, p_i^{n+1}, at the end of the present time step:

$$\left(\frac{AF_i^{n+1}}{\Delta t_{n+1}}\right) p_i^{n+1} + \sum_{j=1}^{NN} p_i^{n+1} BF_{ij}^{n+1} + \nu_i p_i^{n+1} = Q_i^{n+1} \qquad (4.63)$$

$$+ \nu_i p_{BC_i}^{n+1} + q_{IN_i}^{n+1} + DF_i^{(n+1)*} + \left(\frac{AF_i^{n+1}}{\Delta t_{n+1}}\right) p_i^n + \left(CF_i^{n+1}\right)\left(\frac{dU}{dt}_i\right)^n$$

$$i = \overline{1,NN}$$

where the superscript involving (n) or (n+1) indicates level of time evaluation. The term with level (n+1)* indicates that the $(\hat{\rho g})$ term is evaluated at the (n) time level on the first iteration, and at the most recent level on subsequent iterations. The other coefficients are evaluated at the (n+1) time level by projection on the first iteration, and at the most recent level on subsequent iterations.

Boundary conditions, fluid sources and sinks

Specified pressures are obtained through the cellwise addition of a fluid flux, (see Figure 3.7), Q_{BC_i} [M/s] with reference to (4.49):

$$Q_{BC_i}^{n+1} = \nu_i \left(p_{BC_i}^{n+1} - p_i^{n+1}\right) \qquad (4.64)$$

For a cell in which ν_i is specified as a large number, this flux term dominates the fluid mass balance and $p_{BC_i}^{n+1} \simeq p_i^{n+1}$, achieving a specified pressure at the node representing cell i. Note that specified pressure may change each time step. For cells in which pressure is not specified, ν_i is set at zero, and no fluid is added to the cell by (4.64).

Both fluid sources, Q_i^{n+1}, and fluid inflows across region boundaries, $q_{IN_i}^{n+1}$, are specified cellwise. They directly add fluid mass to the node in

cell i. Thus, fluid sources and boundary inflows are indistinguishable in the model. Fluid sources and flows across boundaries are both accounted for by the vector Q_i^{n+1} in SUTRA, and are referred to as fluid sources. Thus the term, $q_{IN_i}^{n+1}$, in (4.63) may be dropped and the definition of Q_i^{n+1} may be generalized to include the boundary flows.

The form of the discretized fluid mass balance implemented in SUTRA is as follows:

$$\sum_{j=1}^{NN} \left[\left(\frac{AF_i\,\delta_{ij}}{\Delta t_{n+1}}\right) + BF_{ij}^{n+1} + \nu_i\delta_{ij}\right] p_j^{n+1} = Q_i^{n+1} + \nu_i p_{BC_i}^{n+1}$$

$$+ DF_i^{(n+1)*} + \left(\frac{AF_i^{n+1}}{\Delta t_{n+1}}\right) p_i^n + \left(CF_i^{n+1}\right)\left(\frac{dU}{dt}_i\right)^n \qquad i = \overline{1,NN} \tag{4.65}$$

wherein δ_{ij} is the Kronecker delta:

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \tag{4.65a}$$

4.5 Numerical Approximation of SUTRA Unified Solute Mass and Energy Balance

The governing equation representing the SUTRA unified energy and solute mass balance (2.52) is modified by the addition of a point source term which arises due to fluid inflows and outflows at points of specified pressure:

$$O_u(U) = \left[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s\right]\frac{\partial U}{\partial t} + \varepsilon S_w \rho c_w \underline{v}\cdot\underline{\nabla}U$$

$$- \underline{\nabla}\cdot\left\{\rho c_w\left[\varepsilon S_w(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}) + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\}$$

$$- Q_p c_w(U^* - U) - \varepsilon S_w \rho \gamma_1^w U - (1-\varepsilon)\rho_s \gamma_1^s U_s - \varepsilon S_w \rho \gamma_o^w - (1-\varepsilon)\rho_s \gamma_o^s$$

$$- Q_{PBC} c_w\left(U_{BC} - U\right) = 0 \tag{4.66}$$

The last term is the solute mass or energy source due to fluid inflow at a point of specified pressure, Q_{PBC} $[M/L^3 \cdot s]$ is the mass fluid source rate given by (4.51), and U_{BC} is the concentration or temperature of the flow. For outflow, $U_{BC} = U$, and the terms goes to zero. Relation (4.66) is numerically approximated in the following sections.

Spatial integration

When the equation for $O_u(U)$ in (4.66) is approximated through nodewise, elementwise and cellwise discretizations, it no longer exactly equals zero. The approximate equation, $\hat{O}_u(U)$, equals a spatially varying residual, $R_u(x,y,t)$, as shown in (3.8). A weighted residual formulation may be written as:

$$\int_V \hat{O}_u(U)\, W_i(x,y)\, dV = 0 \qquad i = \overline{1,NN} \tag{4.67}$$

where $W_i(x,y)$ is the weighting function, chosen to be either the basis function, $\phi_i(x,y)$ or the asymmetric weighting function, $\omega_i(x,y)$, depending on the term of the equation. Relation (4.66) is discretized and the approximation is substituted for $\hat{O}_u(U)$ in (4.67). The resulting set of integral terms is evaluated, one term at a time, in the following paragraphs.

The first term is an integral of the temperature or concentration time derivative:

$$\int_V \left\{ \left[\varepsilon S_w \rho c_w + \hat{(1-\varepsilon)\rho_s c} \right] \frac{\partial U}{\partial t} \right\} \phi_i(x,y)\, dV \tag{4.68}$$

where the term in braces is discretized cellwise, and the weighting function is the basis function, (written in global coordinates). As the term with a carat in braces has constant value over a cell, i, the integral contains only the basis function and equals the cell volume, V_i, according to (3.15). Thus the term is:

$$\left[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s\right]_i \frac{\partial U}{\partial t} i \; V_i \tag{4.69}$$

The second integral is:

$$\int_V \left(\varepsilon S_w \rho c_w \hat{\underline{v}} \cdot \underline{\nabla} U\right) \omega_i(x,y) \; dV \tag{4.70}$$

where the asymmetric weighting function is chosen to allow the use of 'upstream weighting' for this term representing advective transport. 'Upstream weighting' is intended for use only when the finite-element mesh has been designed too coarse for a particular level of dispersive and advective transport. The asymmetric function adds dispersion in an amount dependent on element length in the flow direction. As a result, it changes the parameters and thus changes the physics of the problem being solved. This term is written in general to allow upstream weighting, but simplifies to weighting with a basis function when upstream weight (UP in (4.23) and (4.24)) is set to zero. Thus, in order not to alter the physics for most simulation problems, this term will have symmetric weighting.

The coefficients in this term (except velocity) are evaluated at each Gauss point and are represented depending on nodewise discretization of p and U. Porosity is discretized nodewise. Nodewise discretizations of ε and U are written:

$$\varepsilon(x,y) \simeq \hat{\varepsilon} = \sum_{i=1}^{NN} \varepsilon_i \; \phi_i(x,y) \tag{4.71}$$

$$U(x,y,t) \simeq \sum_{i=1}^{NN} U_i(t) \; \phi_i(x,y) \tag{4.72}$$

The velocity is evaluated at each Gauss point during numerical integration in a particular way that depends on consistent discretization of $\underline{\nabla}p$ and $\rho\underline{g}$ terms in Darcy's law. This consistent approximated velocity is denoted $\hat{\underline{v}}^*$. Thus the term (4.70) is evaluated as:

$$\sum_{j=1}^{NN} U_j(t) \int_x \int_y \left[\hat{\varepsilon}\left(S_w\rho\right) c_w \hat{\underline{v}}^* \cdot \underline{\nabla}\phi_j\right] \omega_i(x,y)\, B(x,y)\, dy\, dx \qquad (4.73)$$

wherein B(x,y) is the nodewise-discretized mesh thickness (4.47). Specific heat, c_w, is a constant.

The third term of (4.67) is:

$$-\int_V \underline{\nabla}\cdot\left\{\rho c_w\left[\varepsilon S_w\left(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}\right) + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\} \phi_i(x,y)\, dV \qquad (4.74)$$

where the basis function weights the integral. Green's Theorem (3.17) is applied to (4.74) resulting in:

$$-\int_\Gamma \left\{\rho c \left[\varepsilon S_w\left(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}\right)\hat{} + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\}\cdot\underline{n}\, \phi_i(x,y)\, d\Gamma$$

$$+\int_V \left\{\rho c \left[\varepsilon S_w\left(\sigma_w \underline{\underline{I}} + \underline{\underline{D}}\right)\hat{} + (1-\varepsilon)\sigma_s \underline{\underline{I}}\right]\cdot\underline{\nabla}U\right\}\cdot\underline{\nabla}\phi_i\, dV \qquad (4.75)$$

where the carat refers to the entire terms in braces. The first term represents the diffusive/dispersive flux of solute mass or energy out across a system boundary in the region of node i. This term is denoted, Ψ_{OUT_i}. An influx would be $-\Psi_{OUT_i}$ or Ψ_{IN_i}. The second term is based on nodewise discretization of U. The coefficients ρ and S_w are evaluated at Gauss points based on nodewise discretization of U and p. Porosity, ε is discretized nodewise as in (4.71), and c_w, σ_w and σ_s are constants. The dispersion tensor, $\underline{\underline{D}}$, is evaluated at each

Gauss point according to equations (2.38) through (2.40b). Velocities used in this evaluation are the consistent values, $\hat{\underline{v}}^*$, and dispersivities, α_L and α_T, are discretized elementwise except that α_L is evaluated at each Gauss point for the anisotrtopic media model. The approximated $\underline{\underline{D}}$ is denoted, $\hat{\underline{\underline{D}}}$ Thus, the term (4.74) is evaluated as:

$$- \Psi_{IN_i} + \sum_{j=1}^{NN} U_j(t) \int_x \int_y \left\{ \rho c_w \left[\hat{\epsilon} S_w \left(\sigma_w \underline{\underline{I}} + \underline{\underline{D}} \right) + (1-\hat{\epsilon})\, \sigma_s \underline{\underline{I}} \right] \right\} \cdot \underline{\nabla} \phi_j \; B(x,y)\, dy\, dx \tag{4.76}$$

The remaining terms in (4.67) are discretized cellwise with the basis function as the weighting function:

$$- \int_V \left[Q_p c_w (\hat{\hat{U}}^* - U) \right] \phi_i(x,y)\, dV = - Q_i c_w \left(U_i^* - U_i \right) \tag{4.77}$$

$$- \int_V \left[\epsilon S_w \rho \gamma_1^w \hat{\hat{U}} \right] \phi_i(x,y)\, dV = - \left[\epsilon S_w \rho \gamma_1^w \right]_i U_i V_i \tag{4.78}$$

$$- \int_V \left[(1-\epsilon) \rho_s \gamma_1^s \hat{\hat{U}}_s \right] \phi_i(x,y)\, dV = - \left[(1-\epsilon) \rho_s \gamma_1^s U_s \right]_i V_i \tag{4.79}$$

$$- \int_V \left[\epsilon S_w \rho \gamma_o^w + \hat{\hat{(1-\epsilon)}}\, \rho_s \gamma_o^s \right] \phi_i(x,y)\, dV = - \left[\epsilon S_w \rho \gamma_o^w + (1-\epsilon) \rho_s \gamma_o^s \right]_i V_i \tag{4.80}$$

$$- \int_V \left[Q_{PBC} c_w \left(\hat{\hat{U}}_{BC} - U \right) \right] \phi_i(x,y)\, dV = - Q_{BC_i} c_w \left(U_{BC_i} - U_i \right) \tag{4.81}$$

where:

$$Q_{BC_i} = \nu_i \left(p_{BC_i} - p_i \right) \tag{4.82}$$

and:

$$\hat{Q}_{PBC} = \sum_{i=1}^{NN} \left(\frac{Q_{BC_i}}{V_i} \right) \tag{4.83}$$

The relation, (4.79), is non-zero only for solute transport and the value of U_s is given for solute trasport by the adsorption isotherms in the form:

$$U_s \equiv C_s = s_L C + s_R \tag{4.84}$$

where s_L and s_R are defined in section 4.7, "Temporal Evaluation of Adsorbate Mass Balance." In the above cellwise relations, c_w, ρ_s, γ_1^w, and γ_1^s are constant, and γ_o^w, γ_o^s, s_L, and s_R may vary cellwise and with time.

By combining and rearranging the evaluations of integrals in (4.67) and the definition (4.84), the following NN spatially discretized weighted residual relations are obtained:

$$AT_i \frac{dU}{dt}_i + \sum_{j=1}^{NN} U_j(t)\, DT_{ij} + \sum_{j=1}^{NN} U_j(t)\, BT_{ij} - (GT_i + G_s TL_i)\, U_i(t) + Q_i c_w U_i(t)$$

$$+ Q_{BC_i} c_w U_i(t) = Q_i c_w U_i^* + Q_{BC_i} c_w U_{BC_i} + \Psi_{IN_i} + ET_i + G_s TR_i \tag{4.85}$$

$$i = \overline{1, NN}$$

where:

$$AT_i = \left[\varepsilon S_w \rho c_w + (1-\varepsilon)\rho_s c_s\right]_i V_i \tag{4.86}$$

$$DT_{ij} = \int_x \int_y \left[\hat{\varepsilon}(S_w \rho) c_w \hat{\underline{v}}^* \cdot \underline{\nabla}\phi_j\right] \omega_i B \, dy \, dx \tag{4.87}$$

$$BT_{ij} = \int_x \int_y \left\{\rho c_w \left[\hat{\varepsilon} S_w \left(\sigma_w \underline{\underline{I}} + \hat{\underline{\underline{D}}}\right) + (1-\hat{\varepsilon})\sigma_s \underline{\underline{I}}\right] \cdot \underline{\nabla}\phi_j\right\} \cdot \underline{\nabla}\phi_i \, B \, dy \, dx \tag{4.88}$$

$$GT_i = \left(\varepsilon S_w \rho \gamma_1^w\right)_i V_i \tag{4.89a}$$

$$G_s TL_i = \left[(1-\varepsilon)\rho_s \gamma_l^s \, s_L\right]_i \, V_i \tag{4.89b}$$

$$G_s TR_i = \left[(1-\varepsilon)\rho_s \gamma_l^s \, s_R\right]_i \, V_i \tag{4.89c}$$

$$ET_i = \left[\varepsilon S_w \rho \gamma_o^w + (1-\varepsilon)\rho_s \gamma_o^s\right]_i \, V_i \tag{4.90}$$

The only integrals requiring Gaussian integration are DT_{ij} and BT_{ij}. Note that these are evaluated in SUTRA subroutine ELEMEN, in an element by element manner. The remaining terms that do not involve Q_{BC} are evaluated cellwise by SUTRA subroutine NODALB. Also note that the flux terms arising from specified pressure (those with Q_{BC}) are evaluated by subroutine BCB.

Temporal discretization and iteration

The time derivative in the spatially discretized and integrated equation is approximated by finite differences:

$$\frac{dU}{dt}_i \simeq \frac{U_i^{n+1} - U_i^n}{\Delta t_{n+1}} \tag{4.91}$$

where:

$$U_i^n = U_i(t^n) \tag{4.92a}$$

$$U_t^{n+1} = U_i(t^n + \Delta t_n) = U_i(t^{n+1}) \tag{4.92b}$$

All terms in (4.85) are evaluated at the new time level, t^{n+1}, except the velocity in (4.87) and the dispersion tensor in (4.88) which involves velocity are lagged on the first iteration. Because coefficients depend on the yet unknown values of p and U at the end of the time step, one or more iterations may be used to solve this non-linear problem. On the first iteration, and when only one iteration per time step is used, coefficients are based on a projected

value of p and U as given by (4.61) and (4.62). On subsequent iterations coefficients are based on the most recent value of p and U. Iterations end when the convergence criteria are satisfied.

On the first iteration, and when only one iteration per time step is used, the velocities are evaluated based on p_i^n, U_i^{n-1} and ρ_i^{n-1}. This is because the pressure gradient in the velocity calculation, $\underline{\nabla} p^n$, is based on pressures calculated when the fluid density was ρ^{n-1}. On subsequent iterations velocities are calculated using the pressure solution for the most recent iteration together with the densities resulting from the previous iteration upon which the most recent pressure solution was based. No spurious velocities, which arise from mismatched p and ρ, are generated this way. The flux term, Q_{BC}, arising from the specified pressures is evaluated on the first iteration at the beginning of the time step in terms of p_i^n and $p_{BC_i}^n$. On subsequent iterations, it is based on the most recent pressure solution and $p_{BC_i}^{n+1}$.

The relations (4.85) may thus be written in a form which allows for solution of concentration or temperature at nodes, U_i^{n+1}, at the end of the present time step:

$$\left(\frac{AT_i^{n+1}}{\Delta t_{n+1}}\right) U_i^{n+1} + \sum_{j=1}^{NN} U_j^{n+1}\, DT_{ij}^{(n+1)*} + \sum_{j=1}^{NN} U_j^{n+1}\, BT_{ij}^{n+1} + \left(GT_i^{n+1} + G_s TL_i^{n+1}\right) U_i^{n+1}$$

$$+ Q_i^{n+1} c_w\, U_i^{n+1} + Q_{BC_i}^{(n+1)*} c_w\, U_i^{n+1} = Q_i^{n+1} c_w\, U_i^{*n+1} + Q_{BC_i}^{(n+1)*} c_w\, U_{BC_i}^{n+1} + \Psi_{IN_i}^{n+1} + ET_i^{n+1}$$

$$+ G_s TR_i^{n+1} + \left(\frac{AT_i^{n+1}}{\Delta t_{n+1}}\right) U_i^n \qquad i=\overline{1,NN} \qquad (4.93)$$

The (n+1)* level indicates that velocity and Q_{BC} are evaluated on the first iteration at the time step (n) and on subsequent iterations, at the most

recent level. Other coefficients are evaluated at the (n+1) time level by projection on the first iteration, and then at the most recent level on subsequent iterations.

Boundary conditions, energy or solute mass sources and sinks

Specified temperatures or concentrations at nodes are obtained numerically at the node k by replacing the k^{th} equation in (4.93) by:

$$U_k^{n+1} = U_{BC_k}^{n+1} \tag{4.94}$$

where $U_{BC_k}^{n+1}$ is the user-specified value of U that node k is to have during time step (n+1). The specified value may change with each time step.

Source boundary conditions for U arise whenever a fluid source Q_i is specified. These may be either point sources of fluid or fluid flows across the boundaries. These fluid inflows must be assigned concentration or temperature values, U_i^{*n+1}, which may change with each time step. Note that these sources are evaluated in SUTRA subroutine NODALB. Outflows of fluid result in the disappearance of the source term from the transport equation because ($U_i^{*n+1} = U_i^{n+1}$) the sink and aquifer have the same U-value.

Source boundary conditions for U may arise at points of specified pressure when an inflow Q_{BC_i} occurs at such a point. A value of U must be specified for such fluid inflows as $U_{BC_i}^{n+1}$. These values may change with each time step. This source term for U disappears for outflow at a point of specified pressure. Note that specified pressure sources are evaluated in SUTRA subroutine BCB.

A source or sink at a boundary due to diffusion or dispersion appears in (4.75):

$$\Psi_{IN_i}^{n+1} = \int_\Gamma \left\{ \rho c_w \left[\varepsilon S_w \left(\sigma_w \underline{\underline{I}} + \underline{\underline{D}} \right) \hat{} + (1-\varepsilon)\, \sigma_s \underline{\underline{I}} \right] \cdot \underline{\nabla} U \right\}^{n+1} \cdot \underline{n}\, \phi_i \, d\Gamma \tag{4.95}$$

where the carat refers to the entire term in braces. For solute transport, this term may represent molecular diffusion and dispersion of solute mass across a boundary. For energy transport, this term represents heat conduction and thermal dispersion across a boundary. This heat or solute flux is a user-specified value which may change each time step. If the term is set to zero, it implies no diffusion and no dispersion across a boundary for solute transport, or for energy transport it implies perfect thermal insulation and no dispersion across a boundary For an open boundary across which fluid flows, this term is <u>not</u> automatically evaluated by SUTRA. If no user-specified value exists at an open boundary, then this term is set to zero. This implicitly assumes that the largest part of solute or energy flux across an open boundary is advectively transported rather than diffusively or dispersively transported. In cases where this assumption is inappropriate, the code may be modified to evaluate this term at the new time level depending on the value of U^{n+1}.

The form of the discretized unified energy and solute mass balance equation which is implemented in SUTRA is as follows:

$$\sum_{j=1}^{NN}\left\{\left(\frac{AT_i^{n+1}\,\delta_{ij}}{\Delta t_{n+1}}\right)+DT_{ij}^{(n+1)*}+BT_{ij}^{n+1}+\left[GT_i^{n+1}+G_sTL_i^{n+1}+\left(Q_i^{n+1}+Q_{BC_i}^{n}\right)c_w\right]\delta_{ij}\right\}U_i^{n+1}$$

$$=c_w\left(Q_i^{n+1}\,U_i^{(n+1)*}+Q_{BC_i}^{n}\,U_{BC_i}^{n+1}\right)+\Psi_{IN_i}^{n+1}+ET_i^{n+1}+G_sTR_i^{n+1}+\left(\frac{AT_i^{n+1}}{\Delta t_{n+1}}\right)U_i^{n} \qquad (4.96)$$

$$i=\overline{1,NN}$$

wherein δ_{ij} is the Kronecker delta.

4.6 Consistent Evaluation of Fluid Velocity

Fluid velocity is defined by equation (2.19) as:

$$\underline{v} = - \left(\frac{\underline{\underline{k}}\, k_r}{\varepsilon\, S_w\, \mu}\right) \cdot \left(\underline{\nabla} p - \rho \underline{g}\right) \tag{4.97}$$

This relation strictly holds true at a point in space. In order for the relation to hold true when discretized, the terms $\underline{\nabla} p$ and $\rho \underline{g}$ must be given the same spatial variability. This avoids generation of spurious velocities which would be caused by local mismatching of the discretized pressure gradient term and density-gravity term. For example, in a hydrostatic system where densities vary spatially, $\underline{\nabla} p = \rho \underline{g}$, to yield a zero vertical velocity. However, if $\underline{\nabla} p$ and $\rho \underline{g}$ do not locally cancel because of the discretization chosen, then erroneous vertical velocities would be generated.

Such an error would occur over an element where $\underline{\nabla} p$ is allowed only a single constant value in a vertical section of the element, but where ρ is allowed to vary linearly in the vertical direction. This would be the case in a standard finite-element approximation wherein both p and U vary linearly in the vertical direction across an element. Linear change in p implies a <u>constant</u> value $\underline{\nabla} p$, while linear change in U implies a <u>linear</u> change in the value of ρ according to (2.3) or (2.4). Thus a standard finite-element approximation over a bi-linear element results in inconsistent approximations in the vertical direction for $\underline{\nabla} p$ and $\rho \underline{g}$: constant $\underline{\nabla} p$ and linearly varying ρ. This inconsistency generates

spurious vertical velocities especially in regions of sharp vertical changes in U. A consistent approximation of velocity is one in which $\underline{\nabla}p$ and $\rho\underline{g}$ are allowed the same spatial variability, and further, are evaluated at the same time level.

A consistent evaluation of velocity is required by the transport solution in (4.87) and also required in the evaluation of the dispersion tensor in (4.88), where velocity is required in each element, in particular, at the Gauss points for numerical integration. Also a consistent evaluation of the $\rho\underline{g}$ term is required for the fluid mass balance solution in the integral shown in (4.56). The values are also required at the Gauss points in each element during numerical evaluation of this integral.

The coefficients for calculation of velocity in (4.97) are discretized as follows: Permeability, $\underline{\underline{k}}$, is discretized elementwise; porosity, ϵ, is discretized nodewise. Unsaturated flow parameters, k_r and S_w, are given values depending on the nodewise-discretized pressure according to relations (2.8) and (2.21). Viscosity is either constant for solute transport or is given values depending on nodewise-discretized temperature according to (2.5).

To complete the discretization of velocity, values in global coordinates at the Gauss points are required for the term $(\underline{\nabla}p - \rho\underline{g})$. A consistent approximation is presented in the remainder of this section for this term based on the fact that this term will be discretized in a consistent manner in global coordinates in an arbitrarily oriented quadrilateral element whenever it is discretized consistently in local element coordinates (ξ,η). Consistent discretization in local coordinates is obtained when the spatial dependence of $\frac{\partial p}{\partial \xi}$ and ρg_ξ is the same, and when $\frac{\partial p}{\partial \eta}$ and ρg_η have the same spatial dependence. Because the discretization for $p(\xi,\eta)$ has already been chosen to be bi-linear, it is the discretization of the $\rho\underline{g}$ term, in particular, which must be adjusted. First,

in the following, a discretization of the $\rho\underline{g}$ term is presented which is presented which is consistent with the discretization of $\underline{\nabla}p$ in local coordinates, and then both $\underline{\nabla}p$ and $\rho\underline{g}$ are transformed to global coordinates while maintaining consistency.

The pressure gradient within an element in local coordinates is defined in terms of the derivatives with respect to the local coordinates:

$$\frac{\partial p}{\partial \xi}(\xi,\eta) = \sum_{i=1}^{4} p_i \frac{\partial \Omega}{\partial \xi} i \tag{4.98a}$$

$$\frac{\partial p}{\partial \eta}(\xi,\eta) = \sum_{i=1}^{4} p_i \frac{\partial \Omega}{\partial \eta} i \tag{4.98b}$$

The summations may be expanded and written in detail by reference to relations (4.9) through (4.12) and (4.1) through (4.4).

A local discretization of $\rho\underline{g}$, with a spatial functionality that is consistent with the local pressure derivatives, (4.98a) and (4.98b) is:

$$(\rho g)_{\xi}(\xi,\eta) = \sum_{i=1}^{4} \rho_i g_{\xi_i} \left| \frac{\partial \Omega}{\partial \xi} i \right| \tag{4.99}$$

$$(\rho g)_{\eta}(\xi,\eta) = \sum_{i=1}^{4} \rho_i g_{\eta_i} \left| \frac{\partial \Omega}{\partial \eta} i \right| \tag{4.100}$$

where the vertical bars indicate absolute value, ρ_i is the value of ρ at node i in the element based on the value of U at the node through relation (2.3) or (2.4), g_{ξ_i} is the ξ-component of $\underline{g}$ at node i, and g_{η_i} is the η-component of $\underline{g}$ at node i. The eight gravity vector componenets at the nodes in each element need be calculated only once for a given mesh and may be saved. This discretization is robust in that it allows both the density and (the direction and) the magnitude of gravity vector components to vary over an element. No particular

significance should be attached to the absolute values of basis function derivatives, except that these happen to give the desired consistent approximations, as is shown shortly.

The gravity vector components in local coordinates at a point in the element are obtained from the global gravity components as:

$$\begin{Bmatrix} g_\xi \\ g_\eta \end{Bmatrix} = \left[J\right] \begin{Bmatrix} g_x \\ g_y \end{Bmatrix} \tag{4.101}$$

where $\left[J\right]$ is the Jacobian matrix defined by (4.25).

The derivatives of pressure in local coordinates (4.98a) and (4.98b), and the consistent density-gravity term components in local coordinates, (4.99) and (4.100), are transformed to global coordinates for use in the evaluation of the integrals they appear in by:

$$\begin{Bmatrix} \frac{\partial p}{\partial x} \\ \\ \frac{\partial p}{\partial y} \end{Bmatrix} = \left[J^{-1}\right] \begin{Bmatrix} \frac{\partial p}{\partial \xi} \\ \\ \frac{\partial p}{\partial \eta} \end{Bmatrix} \tag{4.102}$$

$$\begin{Bmatrix} \hat{(\rho g)}^*_x \\ \hat{(\rho g)}^*_y \end{Bmatrix} = \left[J^{-1}\right] \begin{Bmatrix} (\rho g)_\xi \\ (\rho g)_\eta \end{Bmatrix} \tag{4.103}$$

where $\hat{(\rho g)}^*_x$ and $\hat{(\rho g)}^*_y$ are the consistently discretized density-gravity term components in global coordinates, and $[J]^{-1}$ is the inverse Jacobian matrix defined by (4.29).

The spatial consistency of these approximations may be seen by inspecting their expansions in local coordinates. For example, the ξ-components are:

$$\frac{\partial p}{\partial \xi} = \frac{1}{4}\left[\left(p_2 - p_1 \right)\left(1-\eta\right) + \left(p_3 - p_4 \right)\left(1+\eta\right)\right] \qquad (4.104)$$

$$(\rho g)_\xi = \frac{1}{4}\left[\left(\rho_1 g_{\xi_1} + \rho_2 g_{\xi_2}\right)\left(1-\eta\right) + \left(\rho_3 g_{\xi_3} + \rho_4 g_{\xi_4}\right)\left(1+\eta\right)\right] \qquad (4.105)$$

The terms in parentheses preceeding the terms containing η all have a constant value for the element, and thus the approximations have consistent spatial dependences.

4.7 Temporal Evaluation of Adsorbate Mass Balance

The terms in the unified energy and solute mass balance equation which stem from the adsorbate mass balance require particular temporal evaluation because some are non-linear. The following terms of relation (4.93) are evaluated here: AT_i^{n+1}, GT_i^{n+1}, and ET_i^{n+1}. For solute transport, the coefficient, c_{s_i}, in AT_i^{n+1} (4.86) becomes $\kappa_{1_i}^{n+1}$, according to (2.52b). The relation which defines κ_1 is given by either (1.34c), (1.35c), or (1.36c) depending on the sorption isotherm. The variable, $U_{s_i}^{n+1}$, is expressed in terms of the concentration of adsorbate, $C_{s_i}^{n+1}$, in a form given by (4.84). The parameters in (4.84), s_L and s_R, are defined in this section and are based on either (1.34a), (1.35a) and (1.36a) depending again on the sorption isotherm. The temporal approximations of these parameters are described below for each isotherm.

For linear sorption, all terms and coefficients related to the adsorbate mass are linear and are evaluated at the new time level and strictly solved for at this level:

$$U_{s_i}^{n+1} = C_{s_i}^{n+1} = \chi_1 \rho_o C_i^{n+1} \qquad (4.106a)$$

$$c_{s_i}^{n+1} = \kappa_{1_i}^{n+1} = \chi_1 \rho_o \tag{4.106b}$$

$$s_L = \chi_1 \rho_o \tag{4.106c}$$

$$s_R = 0 \tag{4.106d}$$

For Fruendlich sorption, the adsorbate concentration is split into a product of two parts for temporal evaluation. One part is treated as a first order term as is linear sorption. This part is evaluated strictly at the new time level and solved for on each iteration or time step. The remaining part is evaluated as a known quantity, either based on the projected value of C_i at the end of the time step on the first iteration, or based on the most recent C_i solution on any subsequent iteration.

$$U_{s_i}^{n+1} = C_{s_i}^{n+1} = \left[\left(\chi_1 \rho_o \right)^{\left(\frac{1}{\chi_2}\right)} \left(C_i^{proj} \right)^{\left(\frac{1-\chi_2}{\chi_2}\right)} \right] C_i^{n+1} \tag{4.107a}$$

Also:

$$c_{s_i}^{n+1} = \kappa_{1_i}^{n+1} = \left(\frac{\chi_1}{\chi_2}\right) \rho_o^{\left(\frac{1}{\chi_2}\right)} \left(C_i^{proj} \right)^{\left(\frac{1-\chi_2}{\chi_2}\right)} \tag{4.107b}$$

$$s_L = \left(\chi_1 \rho_o \right)^{\left(\frac{1}{\chi_2}\right)} \left(C_i^{proj} \right)^{\left(\frac{1-\chi_2}{\chi_2}\right)} \tag{4.107c}$$

$$s_R = 0$$

where the coefficient, $\kappa_{1_i}^{n+1}$, is evaluated from the projected or most recent value of C_i, depending on the iteration.

Finally, for Langmuir sorption the form used for the temporal evaluation preserves dependence on a linear relationship to C_i. However, the linear relationship is appropriate only at low solute concentrations. At high concentrations, the adsorbate concentration approaches (χ_1/χ_2). Therefore, two temporal approximations are combined, (one for low C, and one for high C) in a manner depending on the magnitude of concentration. When $(\chi_2\rho_o c)<<1$, the following temporal approximation for low values of C, referred to as C_s^o, is employed:

$$C_s^o = \left(\chi_1\rho_o C^{n+1}\right)\left[1 - \frac{\chi_2\rho_o C^{proj}}{\left(1 + \chi_2\rho_o C^{proj}\right)}\right] \tag{4.108}$$

When $(\chi_2\rho C)>>1$, the following temporal approximation for high C, C_s^∞ is employed:

$$C_s^\infty = \left(\frac{\chi_1}{\chi_2}\right)\left[1 - \frac{1}{\left(1 + \chi_2\rho_o C^{proj}\right)}\right] \tag{4.109}$$

Thus $C_{s_i}^{n+1}$ may be defined:

$$U_{s_i}^{n+1} = C_{s_i}^{n+1} = W_o C_{s_i}^o + W_\infty C_{s_i}^\infty \tag{4.110}$$

where the weights W_o and W_∞, are:

$$W_\infty = \frac{\chi_2\ \rho_o\ C^{proj}}{(1 + \chi_2\ \rho_o\ C^{proj})} \tag{4.111a}$$

$$W_o = 1 - W_\infty \tag{4.111b}$$

By substituting (4.108), (4.109), (4.111a), and (4.111b) into (4.110), the following temporal evaluation of $C_{s_i}^{n+1}$ is obtained after algebraic manipulation:

$$C_{s_i}^{n+1} = \frac{\chi_1 \rho_o C_i^{n+1}}{(1 + \chi_2 \rho_o C_i^{proj})^2} + \frac{(\chi_1 \rho_o C_i^{proj})(\chi_2 \rho_o C_i^{proj})}{(1 + \chi_2 \rho_o C_i^{proj})^2} \quad (4.112a)$$

The coefficient, $\kappa_{1_i}^{n+1}$, is defined as:

$$c_{s_i}^{n+1} = \kappa_{1_i}^{n+1} = \frac{\chi_1 \rho_o}{(1 + \chi_2 \rho_o C_i^{proj})^2} \quad (4.112b)$$

$$s_L = \frac{\chi_1 \rho_o}{\left(1 + \chi_2 \rho_o C_i^{proj}\right)^2} \quad (4.112c)$$

$$s_R = \frac{\left(\chi_1 \chi_2\right)\left(\rho_o C_i^{proj}\right)^2}{\left(1 + \chi_2 \rho_o C_i^{proj}\right)^2} \quad (4.112d)$$

The first term in (4.112a) is solved for on each iteration and the second term is treated as a known. In the above four relations, C_i^{proj} is based on a projection for the first iteration on a time step, and is the most recent value of C_i on subsequent iterations for the time step.

Chapter 5

Other Methods and Algorithms

5.1 Rotation of Permeability Tensor

The aquifer permeability may be anisotropic (as discussed in section 2.2 under the heading "Fluid flow and flow properties," and may vary in magnitude and direction from element to element (as shown in (4.45)). The permeability in each element is completely described by input data values for k_{max}, k_{min} and θ, the principal permeability values and the direction in degrees from the global +x direction to the maximum direction of permeability. The evaluation of integrals (4.55) and (4.56) as well as the velocity evaluation (4.97) require the permeability tensor components in global coordinates as given by (4.45). Thus a rotation of the tensor is required from principal directions (x_p, x_m) to global directions (x,y), as shown in Figure 2.2.

The rotation is given by:

$$\underline{\underline{k}}^L = \underline{\underline{J}}^T \, \underline{\underline{k}}^L_p \, \underline{\underline{J}}^{T^{-1}} \tag{5.1}$$

where

$$\underline{\underline{k}}^L_p = \begin{bmatrix} k^L_{max} & 0 \\ 0 & k^L_{min} \end{bmatrix} \tag{5.2}$$

$$\underline{\underline{J}}^T = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{5.3}$$

$$\underline{\underline{J}}^{T^{-1}} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{5.4}$$

and $\underline{\underline{k}}^L$ is given by (4.45). The result is:

$$k^L_{xx} = k^L_{max} \cos^2\theta + k^L_{min} \sin^2\theta \qquad (5.5a)$$

$$k^L_{yy} = k^L_{max} \sin^2\theta + k^L_{min} \cos^2\theta \qquad (5.5b)$$

$$k^L_{xy} = k^L_{yx} = (k_{max} - k_{min}) \sin\theta \cos\theta \qquad (5.5c)$$

5.2 Radial Coordinates

SUTRA is written in terms of two-dimensional Cartesian coordinates x and y. In general, the two-dimensional numerical methods are applied to Cartesian forms of the governing equations; however, because the mesh thickness, B_i, is allowed to vary from node to node, radial coordinates (cylindical coordinates), r and z are an exact alternate coordinate set.

A function, f(r,z), of radius r, and vertical coordinate z, is integrated over a cylindrical volume as follows:

$$R = \int_z \int_r \int_\theta f(r,z)\ r\ d\theta\ dr\ dz \qquad (5.6)$$

Assuming symmetry with respect to angular coordinate θ (f(r,z) does not depend on θ), the integral becomes:

$$R_r = \int_z \int_r f(r,z)\ (2\pi r)\ dr\ dz \qquad (5.7)$$

This integration may be compared with a general integration of a function g(x,y) in Cartesian coordinates as it is carried out in SUTRA methodology:

$$R_c = \int_y \int_x g(x,y)\ B(x,y)\ dx\ dy \qquad (5.8)$$

Integrals R_r and R_c are exactly analogous if: x≡r, y≡z, and

$$B(x,y) = 2\pi r \tag{5.9}$$

Thus, by a simple redefinition of coordinate names, and by setting the mesh thickness, B, at each node, equal to the circumference of the circle it would sweep out when rotated about the r=0 axis of the cylinder ($B_i=2\pi r_i$), the SUTRA simulation is converted exactly to radial coordinates. Figure 5.1 shows a mesh and the volume it sweeps out when in radial coordinates. Each element becomes a three-dimensional ring when used in radial coordinates.

5.3 Pinch Nodes

Pinch nodes are employed to ease mesh design when large changes in the density of elements are desired over relatively short distances. See Figure 5.2, where pinch nodes are indicated by open nodal dots. This would aid in design of a mesh, for example, in which a large model region is required in order to properly simulate the ground-water flow system. However, only a small portion of this region need be simulated with transport. The fine mesh required in the transport region can be quickly coarsened to the region where only flow is of interest.

Unfortunately, use of pinch nodes tends to increase the band width of the simulation problem although it can significantly decrease the number of nodes in a simulation. Thus with a band-width matrix equation solver, as employed by SUTRA, the use of pinch nodes in a mesh does not always lead to an advantage of decreased computational time. The pinch node option is included, however, as the solver is modular and may be replaced by non-band-width-dependent methods.

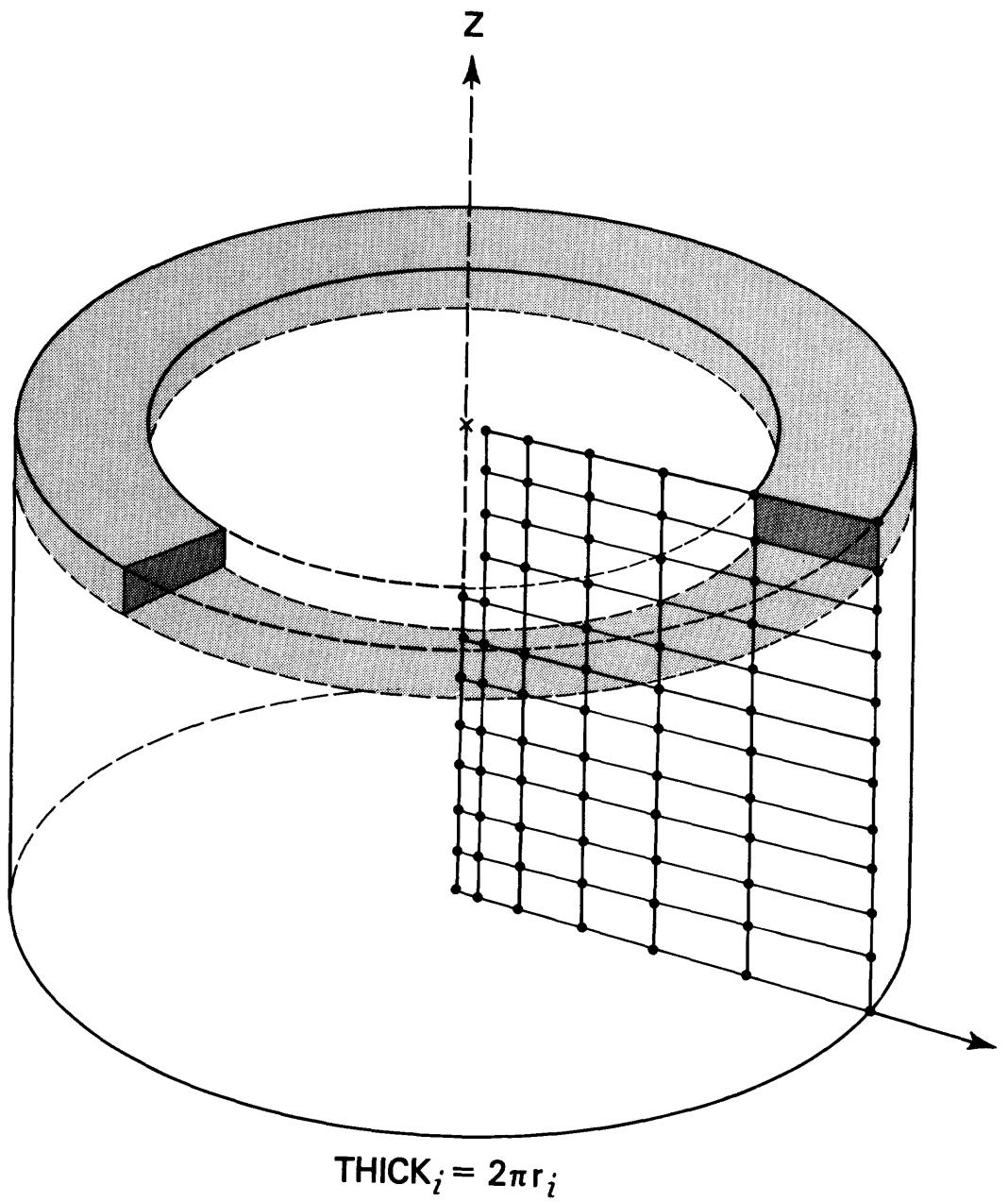


Figure 5.1
Finite-element mesh in radial coordinates.

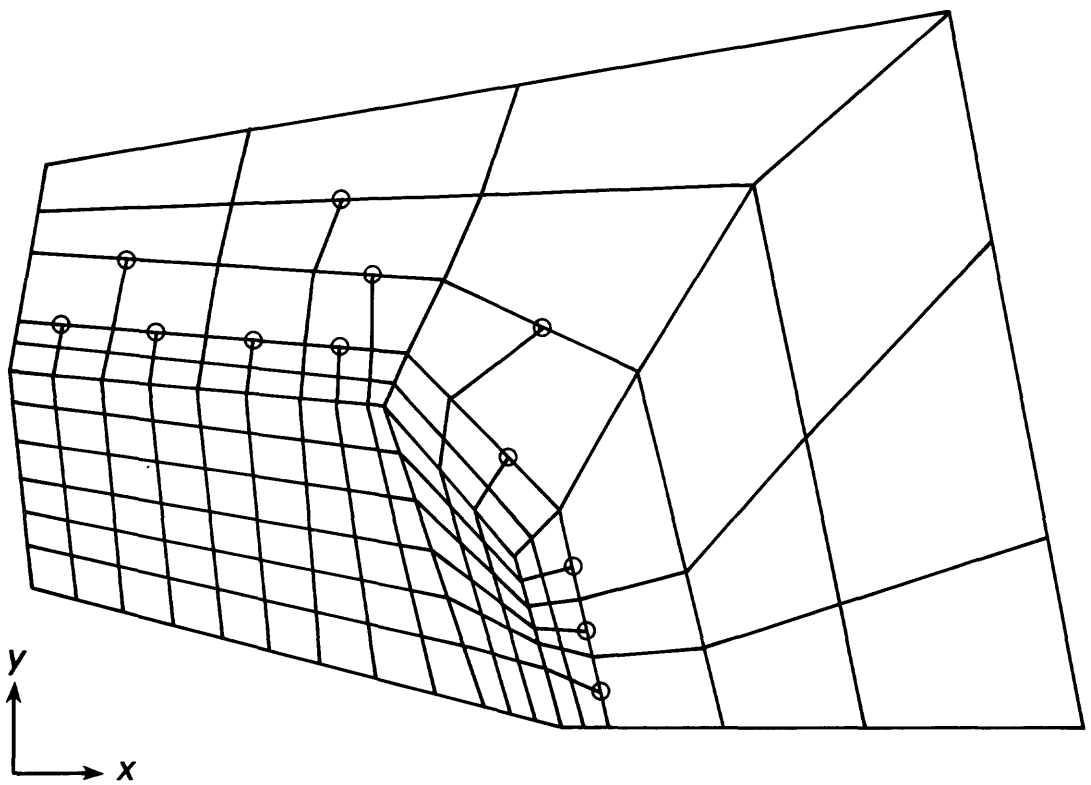


Figure 5.2
Finite-element mesh with pinch nodes.

A pinch node is defined as a node in an element which is located at the mid-point of an element side, as shown in Figure 5.3. Each element has only four real nodes (at the corners) and four basis fuctions associated with these nodes. The pinch node has no basis function assigned to it in the element in which it appears on an element side. Values of variables and coefficients at the pinch node are determined as the average of the values of the real nodes at the end of the element side upon which the pinch node resides. Thus, no sources, sinks, or boundary conditions may be specified at a pinch node. The numerical solution at a pinch node depends entirely on the two nodes at the ends of its side.

Pinch nodes are handled by SUTRA as follows: All elementwise calculations are carried out as though a pinch node were a real node. In fact, each pinch node appears as a corner node in one or more elements. No special treatment is given pinch nodes through the entire matrix assembly process, and they enter the matrix through usual elementwise and nodewise calculations.

Just before solution of the matrix equation, pinch-node conditions are imposed on the matrix equation. For the pinch node, k, the right hand side of the equation for node k is set to zero. The row of the final coefficient matrix for node k is changed to all zeroes, except for two coefficients. These are in the two matrix columns related to the nodes at the ends of the element side upon which pinch node k resides. They are set to a value, 0.50, and the coefficent on the matrix diagonal (with subscript, kk) is set to a value, -1.0. This sets an equation for pinch node k as follows:

$$p_k^{n+1} = \frac{1}{2}\left(p_r^{n+1} + p_s^{n+1}\right) \qquad (5.10a)$$

$$U_k^{n+1} = \frac{1}{2}\left(U_r^{n+1} + U_s^{n+1}\right) \qquad (5.10b)$$

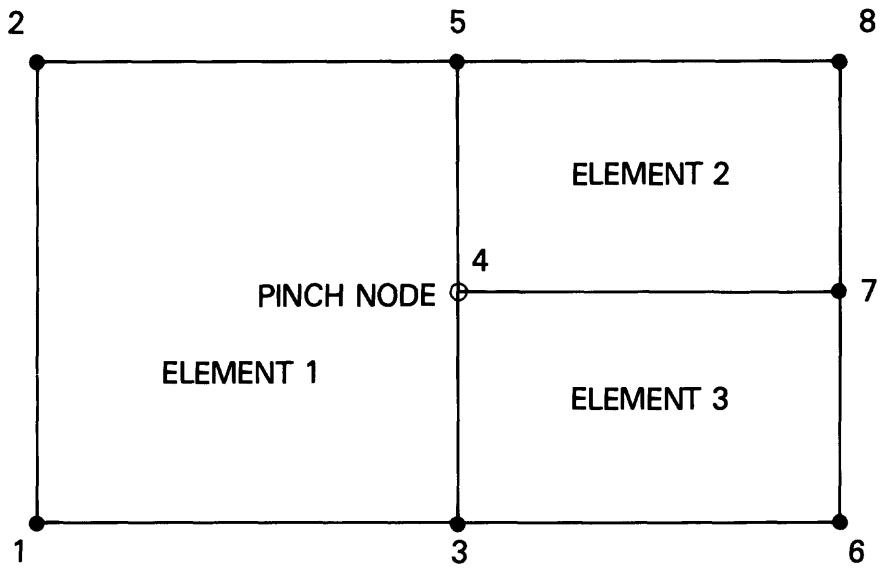


Figure 5.3
Detail of mesh with a pinch node.

where subscripts r and s refer to the nodes at the ends of the pinch node element side.

Pinch nodes are specified in the data set containing the nodal incidence list for elements and the order of specification is related to the local element node numbering scheme as defined by Figure 5.4.

5.4 Solution Sequencing

On any given time step, the matrix equations are created and solved in the following order: (1) the matrix equation for the fluid mass balance is set up, (2) the transport balance matrix equation is set up, (3) pressure is solved for, and (4) concentration or temperature are solved for. Both balances are set up on each pass such that the elementwise calculations only need be done once per pass. However, SUTRA allows the p or U equation to be set up and solved only every few time steps in a cyclic manner based on parameters NPCYC and NUCYC. These values represent the solution cycle in time steps. For example, setting up and solving for both p and U each time step (NPCYC=NUCYC=1):

```
time step: 1 2 3 4 5 6 7 . . . .
          {p p p p p p p
solve for:{
          {U U U U U U U
```

or solving for p every third time step and for U each time step (NPCCYC=3 and NUCYC=1):

```
time step: 1  2  3  4  5  6  7  8  9  10 11 12 13 . . . .
          {p  .  p  .  .  p  .  .  p  .  .  p  .
solve for:{
          {U  U  U  U  U  U  U  U  U  U  U  U  U
```

However, either of p or U must be solved for on each time step and therefore

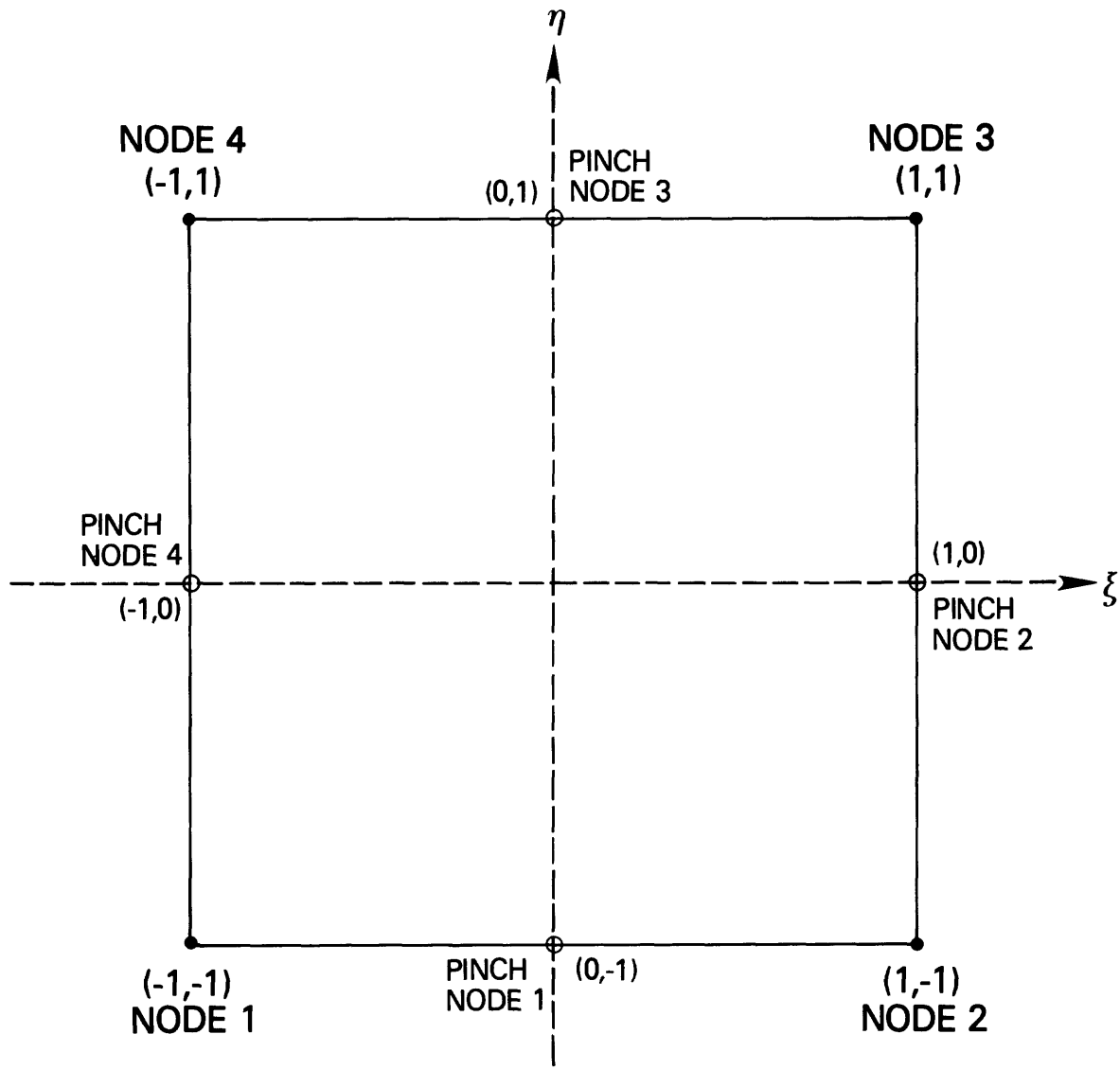


Figure 5.4
Finite element in local coordinates (ξ,η) with pinch nodes.

either NPCYC or NUCYC must be set to one.

For a simulation with steady state flow, the sequencing is:

```
time step: 0 1 2 3 4 5 . . . .
          {p . . . . .
solve for:{
          {U U U U U U
```

For steady flow and transport:

```
time step: 0  1
          {p  .
solve for:{
          {.  U
```

The only exception to the cycling is that for non-steady cases, both unknowns are solved for on the first time step, as shown in the case for NPCYC=3, NUCYC=1, above.

It is computationally advantageous to allow a matrix equation solution for U by back-substitution only, saving both equation construction and matrix decomposition steps. This is begun on the second time step solving for U only after the step on which both p and U are solved for. In order to do this the matrix coefficients including the time step must remain constant. Thus, non-linear variables and fluid velocity are held constant with values used on the first time step for U after the step for p and U. For example, when NPCYC=1, NUCYC=6:

```
time step: 1   2   3   4   5   6   7   8   9   10  11  12
          {p   .   .   .   .   p   .   .   .   .   .   p
solve for:{
          {U   U   U   U   U   U   U   U   U   U   U   U

             |constant values|     | constant  values |
                 | back      |         |   back       |
                 | substitute|         |   substitute |
```

A pressure-only solution may be obtained with NPCYC=1, and NUCYC=(number larger than the number of time steps). Note that p and U solutions must be set to occur on time steps when relevant boundary conditions, sources or sinks are set to change in value.

5.5 Velocity Calculation for Output

The velocities employed in the numerical solution of fluid mass, and solute mass or energy balances are those calculated at the Gauss points in each element (as described in section 4.6 "Consistent Evaluation of Fluid Velocity.") For purposes of output, however, only one velocity value per element is made available. This is the velocity at the element centroid as shown in Figure 5.5. The centroid is defined as the point in the element where the lines connecting the mid-point of opposite sides intersect.

The velocity at the centroid of an element is calculated by taking the average of the four global x-components of velocity at the Gauss points, as well as the average of the four global y-components of velocity at the Gauss points, and by constructing a velocity vector from these averaged components. This process gives the "true" velocity at the centroid that would be calculated employing the consistent velocity approximation evaluated at this point in the element. This may be seen by setting $\xi=\eta=0$ in (4.104) and (4.105).

5.6 Budget Calculations

A fluid mass and solute mass or energy budget provides information on the quantities of fluid mass and either solute mass or energy entering or exiting

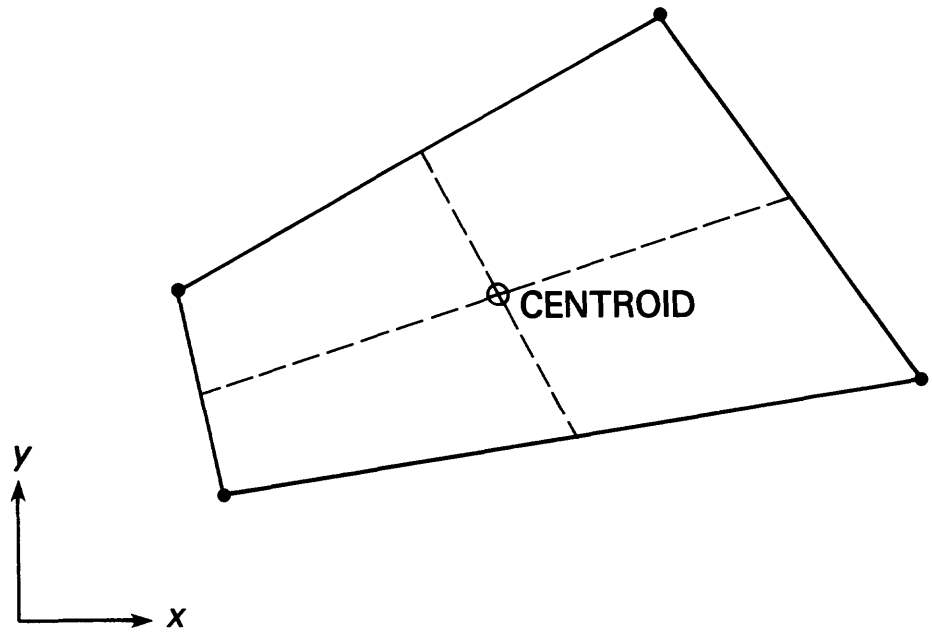


Figure 5.5
Finite element in global coordinates (x,y) with element centroid.

the simulated region. It is not intended as a check on numerical accuracy, but rather as an aid in interpreting simulation results.

The fluid budget is calculated based on the terms of the integrated-discretized fluid mass balance, (4.52), as approximated in time according to (4.65). After the solution to a time step makes available p_i^{n+1} and U_i^{n+1}, the time derivatives of these, $\frac{dp}{dt}i$ and $\frac{dU}{dt}i$, are calculated according to (4.57) and (4.91).

The total rate of change in stored fluid mass in the region due to pressure changes over the recent time step is:

$$\sum_{i=1}^{NN} AF_i^{n+1} \frac{dp_i}{dt} \qquad [M/s] \qquad (5.11)$$

where AF_i is defined in (4.53), and the total rate of change in stored fluid due to changes in concentration or temperature is:

$$\sum_{i=1}^{NN} CF_i^{n+1} \frac{dU_i}{dt} \qquad [M/s] \qquad (5.12)$$

where CF_i as defined in (4.54).
The sum of (5.11) and (5.12) gives the <u>total</u> rate of change of fluid mass in the entire region.

Fluid sources, Q_i^{n+1}, may vary with time and those that do vary are reported by the budget at each source node. The sum of Q_i^{n+1}:

$$\sum_{i=1}^{NN} Q_i^{n+1} \qquad [M/s] \qquad (5.13)$$

gives the total rate of fluid mass change due to all sources and sinks of fluid mass, as well as to specified fluxes across boundaries. Fluid sources due to specified pressure conditions, $Q_{BC_i}^{n+1}$, usually vary with time and are

also reported by the budget at each node. This source is calculated at each node from (4.64). The sum of $Q_{BC_i}^{n+1}$:

$$\sum_{i=1}^{NN} Q_{BC_i}^{n+1} \qquad [M/s] \tag{5.14}$$

gives the total rate of fluid mass change in the entire region due to inflows and outflows at all specified pressure nodes.

The sum of (5.13) and (5.14) should be close to the value given by the sum of (5.11) and (5.12). These may be expected to match better when iterations have been used and convergence achieved, as the budget is calculated for a time step with only one iteration with the (n+1) time level values of non-linear coefficients, and the solution was obtained with coefficients based on projected values of p and U.

The solute mass or energy budget is calculated based on the terms of the integrated-discretized balance, (4.85), as approximated in the time according to (4.93). The total rate of change in stored solute mass or energy in the region over the recently computed time step is:

$$\sum_{i=1}^{NN} AT_i^{n+1} \frac{dU_i}{dt} \qquad [M_s/s \text{ or } E/s] \tag{5.15}$$

where AT_i^{n+1} is calculated from (4.86) using U_i^{n+1} in all coefficients requiring a U value (including adsorption isotherms for $c_s = \kappa_1$). In reporting this portion of the budget, a separate value is given for the sum of the portion stemming from $(\varepsilon S_w \rho c_w)$ and for $(1-\varepsilon)\rho_s c_s$. The former sum relates to rate of solute mass or energy change in the fluid, and the latter relates to change in the solid-immobile portion.

The total rate of first-order solute mass production in the fluid is calculated as:

$$\sum_{i=1}^{NN} GT_i^{n+1} U_i^{n+1} \qquad [M/s] \qquad (5.16a)$$

and at the rate of first-order adsorbate production is calculated as:

$$\sum_{i=1}^{NN} G_s TL_i^{n+1} U_i^{n+1} + G_s TR_i^{n+1} \qquad [M/s] \qquad (5.16b)$$

where GT_i and $G_s TL_i$ and $G_s TR_i$ are defined by 4.89a, 4.89b and 4.89c and all isotherms are based on U_i^{n+1}. Fluid and adsorbate rates are reported separately by the budget. These terms have no analogy for energy transport. The terms of zero-order production of solute and adsorbate mass or energy production in the fluid and solid matrix are:

$$\sum_{i=1}^{NN} ET_i^{n+1} \qquad [M_s/s \text{ of } E/s] \qquad (5.17)$$

where ET_i is defined by (4.90) and the fluid and immobile phase production rates are reported separately by the budget.

Solute mass and energy sources and sinks due to inflowing or outflowing fluid mass may vary with time and are reported by the budget at each fluid source node and at each specified pressure node. These are separately summed for the entire region:

$$\sum_{i=1}^{NN} Q_i^{n+1} c_w U_i^{*n+1} \qquad [M_s/s \text{ or } E/s] \qquad (5.18)$$

$$\sum_{i=1}^{NN} Q_{BC_i}^{n+1} c_w U_{BC_i}^{n+1} \qquad [M_s/s \text{ or } E/s] \qquad (5.19)$$

Where U_i^{*n+1} and $U_{BC_i}^{n+1}$ take on the user-specified values of U for fluid inflows, and the U value of the ambient system fluid for outflows.

These sums give the total rate of change of solute mass or energy in the entire system due to these fluid sources and sinks.

Finally the diffusive-dispersive sources of solute mass or energy are summed for the entire system and are also reported by node as they may vary with time:

$$\sum_{i=1}^{NN} \psi_{IN_i}^{n+1} \qquad [M_s/s \text{ or } E/s] \qquad (5.20)$$

The sum of (5.16a), (5.16b), (5.17), (5.18), (5.19) and (5.20) should be close to the value given by (5.15). These values may be expected to match best when iterations have been used and convergence achieved, as the budget is calculated for a time step with only one iteration with all information at the (n+1) time level, and the solution was obtained using non-linear coefficients based on projections of p and U.

5.7 Program Structure and Subroutine Descriptions

SUTRA is structured in a modular, top-down programming style that allows for code readability, ease in tracing logic, and hopefully, ease in eventual modifications. Each subroutine carries out a primary function that is clearly distinguished from all other program functions. User-required program changes are limited to: 1) dimensioning three storage arrays in the main routine, and

2) coding portions of a subroutine which is used to control time-dependent sources and boundary conditions (when they are used) and a subroutine which sets the unsaturated flow functions when unsaturated flow is simulated. The code consists of approximately 3000 statements and includes one main program and 24 subroutines. The program is commented to aid in tracing logic.

SUTRA is written in FORTRAN-77; however, few structures are used which are not compatible with FORTRAN-66. Modifications of the code required to compile in FORTRAN-66 would not be major.

The code runs accurately when it employs "double-precision" real variables (64 bit words with 47 bit mantissa) with a precision of about 15 significant figures, and 32 bit word integer variables. Should the code require modification to run on machines with other word lengths or other bit to byte ratios, the number of significant figures in a real variable should be preserved, if not increased.

Input and output is also somewhat modularized. Input is through Fortran unit numbers 5 and 55. Unit-55 contains only data on initial conditions for a simulation at the nodes for p and U. Unit-5 contains all other data required for a simulation. Output is to Fortran unit numbers 6 and 66. Unit-66 receives the result of the final time step in a format equivalent to that of Unit-55, for later use as the intial conditions file if the simulation is to be restarted. Unit-6 receives all other simulation output usually to be printed on a line printer (as shown in Figure 5.6).

The main logic flow of the program is straightforward. A schematic diagram of the code is shown in Figure 5.7. The main routine sets up dimensions and calls the main control routine, SUTRA, which cycles the program tasks by calling most of the remaining subroutines in sequence. Subroutines are named to describe their main function. A description of each subroutine is given in the following sections.

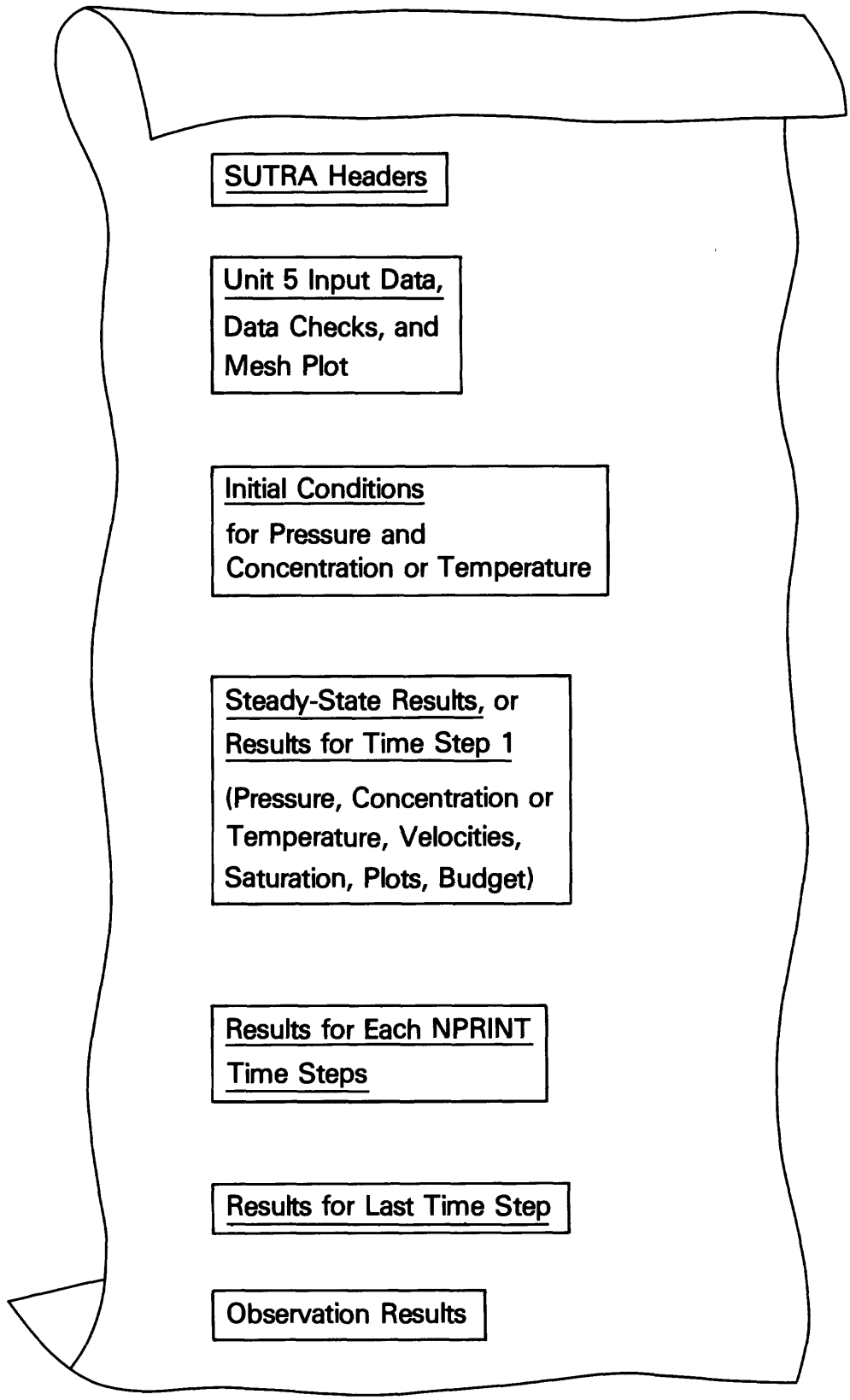


Figure 5.6
Schematic of SUTRA output.

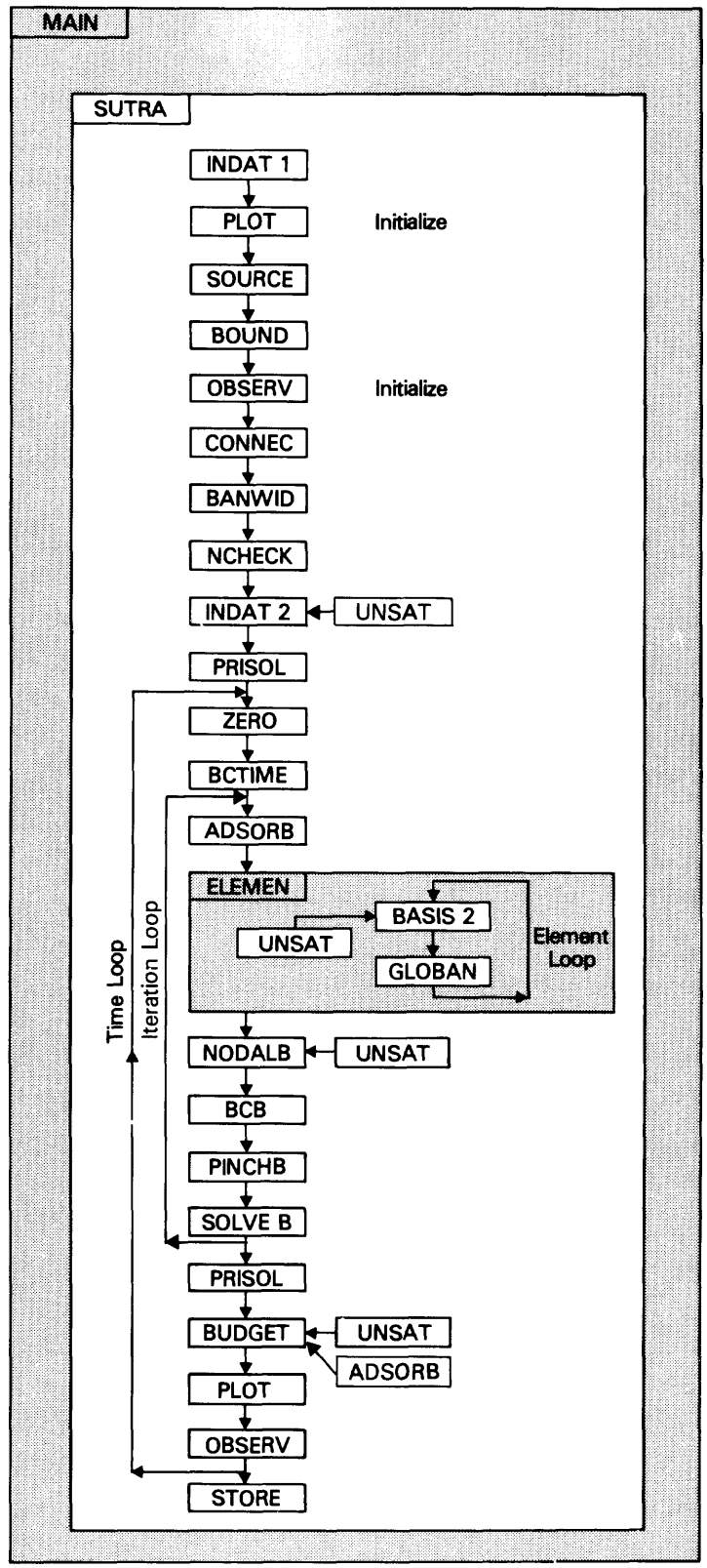


Figure 5.7
SUTRA logic flow.

Main Program

-Purpose:

1) To dimension and allocate space for the main storage arrays.

2) To divide the storage arrays into their component arrays. (Set up pointers.)

3) To start and stop the simulation.

-Calls to:

SUTRA

-Description:

The main routine has three arrays that must be user-dimensioned: RM, RV and IMV. These are used for dynamic storage allocation and they contain almost all of the values required for SUTRA simulation. RM contains real matrices, RV contains real vectors, and IMV contains integer matrices and vectors. The dimensions required for RM, RV and IMV are RMDIM, RVDIM and IMVDIM, where the actual values are given in section 7.3, "Program Dimensions."

After reading the actual Unit-5 input data for the variables listed above, the main routine sets up pointers which allocate the correct amount of space for each of the component arrays contained in the storage arrays. The pointers point to the position in the storage array of the starting element of each component array. The starting elements are passed to subroutine SUTRA as calling arguments Additional arrays which may be required by any modifications to SUTRA are added a

the bottom of the appropriate pointer lists in the call statement and in the pointer calculations. The values of NNV or NEV may need to be increased, and the commented record of calculation of dimensions of storage arrays at the top of the routine should be increased accordingly.

Subroutine SUTRA

-Purpose:

1) To act as primary control on SUTRA simulation, cycling both iterations and time steps.
2) To sequence program operations by calling subroutines for input, output and most program calculations.
3) To carry out minor calculations.

-Called by:

Main routine

-Calls to:

INDAT1, PLOT, SOURCE, BOUND, OBSERV, CONNEC, BANWID, NCHECK, INDAT2, PRISOL, ZERO, BCTIME, ADSORB, ELEMEN, NODALB, BCB, PINCHB, SOLVEB, BUDGET, STORE.

-Description:

Subroutine SUTRA receives pointers for all actual arrays and vectors which are dynamically allocated space by the main routine. These arrays are

dummy dimensioned to actual sizes required for the simulation. Subroutine SUTRA initializes some constants and directs the reading of Unit-5 input data by calls to INDAT1, PLOT, SOURCE, BOUND, OBSERV, and CONNEC. It calls for band-width calculation (BANWID) and mesh data checks (NCHECK). The call to PLOT (after INDAT1) also plots the mesh. Then subroutine SUTRA directs a call to INDAT2 to read initial conditions from Unit-55, and calls PRISOL to print the initial conditions.

The subroutine decides on cycling parameters if steady state pressures will be calculated, and calls ZERO to initialize arrays. For transient pressure solution steps, time-step cycling parameters are set and a decision is made as to which (or both) of p and U will be solved for on this time step. The decision depends on NPCYC and NUCYC, and subroutine SUTRA sets the switch, ML, as follows:

$$ML = \begin{cases} 0 & \text{solve for both p and U} \\ 1 & \text{solve for p only} \\ 2 & \text{solve for U only} \end{cases}$$

The switch for steady state flow is ISSFLO, which is set as follows:

$$ISSFLO = \begin{cases} 0 & \text{steady flow not assumed} \\ 1 & \text{steady flow assumed, before pressure time step} \\ 2 & \text{steady flow assumed, after beginning of pressure time step} \end{cases}$$

Note that time step number, IT, is set to zero for the steady p solution, and increments to one for the first transport time step.

Subroutine SUTRA increments the simulation clock, TSEC, to the time at the end of the new time step, and shifts new vectors to previous level vectors which begins the time step. BCTIME is called to set time-dependent sources and boundary conditions if such exist. ADSORB is called if sorption is required. The element

by element calculations required to construct the matrix equations are carried out by a call to ELEMEN. NODALB is called to carry out nodewise and cellwise calculations for the global matrices. BCB is called to modify the matrix equations for boundary conditions, and PINCHB is called to implement any pinch nodes in the matrices.

SOLVEB is called for p and or U solution (depending on the value of ML), and if iterations are underway, convergence is checked. If iterations are continued, control switches back to the step which shifts new to old vectors, and the sequence of calls is repeated. If no more iterations are required, SUTRA may call PRISOL and PLOT to print and plot results if these are requested on the present time step. OBSERV is called to remember values at observation nodes if any exist. BUDGET is called if requested output should occur this time step.

If more time steps are to be undertaken, control switches back to the step which initializes arrays, and continues down from that point. If the simulation is complete, STORE is called if the store option has been selected to set up a restart file in Unit-66. OBSERV is called to print any observations that were taken. At this point, control returns to the main routine.

Subroutine INDAT1

-Purpose:

1) To read simulation and mesh data from the Unit-5 data file, and print out this information.
2) To initialize some variables and carry out minor calculations.

-Called by:

SUTRA

-Description:

INDAT1 reads a portion of the Unit-5 input data file, ending with the elementwise data set. Most information is printed on the Unit-6 data file after reading, the amount of output depends on the user choice of long or short output format. Scale factors are multiplied with appropriate input data. Calculations are carried out for a thermal conductivity adjustment and for determination of $\underline{\underline{k}}$ matrix components of $\underline{\underline{k}}$ in each element from k_{max}, k_{min}, and θ.

Subroutine PLOT

-Purpose:

To provide maps on printer output paper of the finite-element mesh, pressure values at nodes, and U values at nodes.

-Called by:

SUTRA

-Description:

PLOT is called once for initialization to read plot set-up data from Unit-5, and to set up a plot of the mesh. PLOT is then called to plot the mesh. PLOT is called on each time step in which output is produced, once each for p for U, if these plots have been requested.

The printer plot either fits the longer plot direction across the output page, or along the output page, depending on the user choice. The plot is self-scaled to page size, and different scales may be chosen by the routine along and across the page. A blank border one tenth of the maximum x and y

range surrounds the plotted region. Three figures of the solution value are plotted at each nodal location.

PLOT begins by ordering the nodes by plot line and saves the ordered results in XX, YY, and INDEX during the initialization call. Certain nodes fall in each line of the plot. During actual plotting, PLOT starts with nodes in the top plot line and places the values to be plotted in the proper position in the line. The line is then printed. This is repeated for each line of the plot.

Subroutine SOURCE

-Purpose:

1) To read source node numbers and source values for fluid mass sources and boundary fluxes and for diffusive and productive U sources, as well as fluxes of U at boundaries; to check the data, and to print information.
2) To set up pointer arrays which track the source nodes for the simulation.

-Called by:

SUTRA

-Description:

SOURCE reads and organizes, checks and prints information on source nodes for fluid mass, and for sources of solute mass or energy. Fluid mass source information read is node number, mass source rate, and U value of any inflowing

fluid at this node. If there are NSOP fluid source nodes, the node numbers become the first NSOP values in vector IQSOP. The rates are entered in the element corresponding to the nodes at which they are defined in vectors QIN and UIN which are NN long. The source information for U read is node number and solute mass or energy source rate. If there are NSOU source nodes for U, the node numbers become the first NSOU values in IQSOU. Vector QUIN is NN long and contains the source rates in numerical order by node. Counts are made of each type of source and are checked against NSOP and NSOU for correctness. A blank (zero) node number ends the data set for QIN and then for QUIN. One blank element is left at the end of IQSOP and IQSOU so that a dimension of one is obtained even when no source nodes exist. These arrays are used primarily in NODALB and BUDGET.

Subroutine BOUND

-Purpose:

1) To read specified pressure node numbers and pressure values, check the data, and print information.
2) To read specified concentration or temperature node numbers and the values, to check the data, and print information.
3) To set up pointer arrays which track the specified p and U nodes for the simulation.

-Called by:

SUTRA

-Description:

BOUND reads and organizes, checks and prints information on specified p nodes and for specified U nodes. Pressure information read is node number, pressure value and U value of any inflow at this node. If there are NPBC specified pressure nodes, the above information becomes the first NPBC values in vectors IPBC, PBC and UBC. Specified U information read is node number and U value. If there are NUBC specified concentration nodes, the above information begins in the (NPBC+1) position of IUBC and UBC, and ends in the (NUPBC+NUBC) position of UBC and IUBC. This is shown below:

```
IPBC  ( 1 2 3 4 5 6 7 8 9 10 11 )
      ( x x x x x x             )

PBC   ( 1 2 3 4 5 6             )
      ( x x x x x x             )

UBC   ( x x x x x x y y y y     )
      ( 1 2 3 4 5 6 7 8 9 10 11 )

IUBC  (           y y y y     )
      ( 1 2 3 4 5 6 7 8 9 10 11 )
```

where x refers to specified p information, and y refers to specified U information.

Counts are made of each type of specification and are checked against NPBC and NUBC for correctness. A blank (zero) node number ends the data set for p and then for U. One blank element is left at the end of each of these arrays in case there are no specified p or U nodes. The first NPBC elements of IUBC and UBC are blank. These arrays are used primarily by subroutines BCB and BUDGET.

Subroutine OBSERV

-Purpose:

1) To save p and U values at chosen observation nodes as a function of the time.

2) To report the observations after the simulation has been completed.

-Called by:

SUTRA

-Description:

On an initialization call from SUTRA, OBSERV reads observation node numbers and observation cycle, NOBCYC, in time steps from Unit-5 and outputs these values. Every NOBCYC time steps, when SUTRA calls OBSERV after a solution, OBSERV saves the current elapsed time, and p and U values at all observation nodes. When the simulation is completed, OBSERV is called to output the stored lists of: time step, elapsed time, p, and U.

Subroutine CONNEC

-Purpose:

1) To read, output, and organize node incidence data.

2) To read, output, and organize pinch-node incidence data.

-Called by:

SUTRA

-Description:

CONNEC reads the nodal incidence list which describes how nodes are connected. The data is organized as array, IN, which contains the counter-clockwise-ordered set of four node numbers in each element in order of element number. Thus the ninth through twelfth values in IN are the four nodes in element number three.

For an element with one or more pinch nodes, the pinch node numbers are entered in the first column of array IPINCH, and the node numbers at the ends of the side on which the pinch node resides are entered in columns two and three of IPINCH.

IPINCH is used in subroutine PINCH, and NCHECK. IN is used in BANWID, ELEMEN, and GLOBAN.

Subroutine BANWID

-Purpose:

To calculate the band width of the mesh and check the value specified by the user.

-Called by:

SUTRA

-Description:

BANWID checks the array, IN, in all elements for the maximum difference in node numbers contained in an element. This value, NDIFF, is used to calculate

actual bandwidth, NBL, which is compared with the user-specified, NBI. IF NBL>NBI, the values are printed and the simulation is halted for corrections.

Subroutine NCHECK

-Purpose:

To check that pinch nodes are neither assigned sources, nor have specified p or U.

-Called by:

SUTRA

-Description:

NCHECK compares the list of pinch node numbers with the list of source nodes, specified pressure nodes and specified U nodes. Any matches result in a printed report and the simulation halts.

Subroutine INDAT2

-Purpose:

1) To read initial conditions from Unit-55.

2) To initialize some arrays.

-Called by:

SUTRA

-Calls to:

UNSAT

-Description:

INDAT2 is the second major input data routine. It reads the data file, Unit-55, which contains initial conditions for p and U. The warm-start section reads initial conditions and parameter values of a previous time step, all of which must have been stored by subroutine STORE on a previous simulation. For a cold-start, INDAT2 reads only initial p and initial U. INDAT2 calls UNSAT for calculation of initial saturation values, on a cold start.

Subroutine PRISOL

-Purpose:

To output the following to Unit-6:

Initial conditions

Pressure solutions

Saturation values

Concentration and temperature solutions

Steady-state pressure solution

Fluid velocities (magnitude and direction)

-Called by:

SUTRA

-Description:

PRISOL is the main SUTRA output routine and is used for printing solutions.

Subroutine ZERO

-Purpose:

To fill a real array with a constant value.

-Called by:

Various routines

-Description:

ZERO fills an entire array with a specified value. This routine may be replaced with a machine-dependent assembly language routine in order to maximize efficiency.

Subroutine BCTIME

-Purpose:

A user-programmed routine in which time-dependent sources and boundary conditions are specified.

-Called by:

SUTRA

-Description:

BCTIME is called on each time step when a time-dependent source or boundary condition is specified by the user. It allows the value of a source or boundary condition to be changed on any or all time steps.

BCTIME is divided into four sections. The first section allows the user to specify either time-dependent pressure and concentration or temperature of an inflow, or both, at specified pressure nodes (PBC or UBC). The second section allows user specification of time-dependent U at specified concentration/temperature nodes. The third section allows user specification of time-dependent fluid source or source concentration/temperature. The fourth section allows user-specification of time-dependent solute mass or energy source.

The current time step number, IT, and current time (at the end of the present time step) in various units are available for use in the user-supplied programming. The user may program in any convenient way through data statements, calls to other programs, logical structures, 'read' or 'write' statements, or other preferred methods of specifying the time variability of sources or specified p and U conditions. More information may be found in section 7.5, "User-Supplied Programming."

Subroutine ADSORB

-Purpose:

To calculate and supply values from adsorption isotherms to the simulation.

-Called by:

SUTRA

-Description:

ADSORB calculates the sorption coefficient, $\kappa_{l_i}^{n+1}$, (called CS1), and also s_L (called SL) and s_R (called SR) and SR which are used in calculating) adsorbate concentrations, U_s, depending on the particular isotherm chosen: linear, Freundlich or Langmuir. The calculations are based on the description given in section 4.7, "Temporal Evaluation of Adsorbate Mass Balance." ADSORB is called once per time step for U, when sorption is employed in the simulation.

Subroutine ELEMEN

-Purpose:

1) To carry out all elementwise calculations required in the matrix equations.
2) To calculate element centroid velocities for output.

-Called by:

SUTRA

-Calls to:

BASIS2, GLOBAN

-Description:

ELEMEN undertakes a loop through all the elements in a mesh. For each element, subroutine BASIS2 is called four times, once for each Gauss point.

BASIS2 provides basis·function information, and values of coefficients and velocities at each Gauss point, all of which is saved by ELEMEN for use in calculations for the present element.

Gaussian integration (two by two points) as described in section 4.3, is carried out for each integral in the fluid mass balance ((4.55) and (4.56)), and for each integral in the unified energy and solute mass balance ((4.87) and (4.88)). The portion of cell volume within the present element for node I, VOLE(I), is calculated with the fluid balance integrals. The values of the integrals are saved either as four-element vectors or as four-by-four arrays. Separate (nearly duplicate) sections of the integration code employ either basis functions for weighting or asymmetric weighting functions.

The vectors and arrays containing the values of integrals over the present element are passed to subroutine GLOBAN in order to add them to the global matrix equation (assembly process).

Subroutine BASIS2

-Purpose:

To calculate values of basis functions, weighting functions, their derivatives, Jacobians, and coefficients at a point in a quadrilateral element.

-Called by:

ELEMEN

-Calls to:

UNSAT

-Description:

BASIS2 receives the coordinates of a point in an element in local coordinates (ξ,η), denoted (XLOC,YLOC) in the routine. At this point, BASIS2 determines the following: values of the four basis functions and their derivatives in each local coordinate direction, elements of the Jacobian matrix, the determinant of the Jacobian matrix, elements of the inverse Jacobian matrix, and if required, four values of the asymmetric weighting function (one for each node) and their derivatives. Also, the derivatives are transformed to global coordinates and passed out to ELEMEN. Values of nodewise-discretized parameters are formed at this location in the element, as are values of local and global velocity. Values of parameters dependent on p or U are calculated at this location. Unsaturated parameters are obtained by a call to UNSAT. The calculations are based on sections, 4.1 "Basis and Weighting Functions", 4.2 "Coordinate Transformations," and 4.6 "Consistent Evaluation of Fluid Velocity."

Subroutine UNSAT

-Purpose:

A user-programmed routine in which unsaturated flow functions are specified.

-Called by:

INDAT2, BASIS2, NODALB, BUDGET

-Description:

UNSAT is called by INDAT2 to calculate initial saturations at nodes, by BASIS2 at each Gauss point in each element during numerical integration, by NODALB for each cell, and by BUDGET for each cell. It allows the user to specify the functional dependence of relative permeability on saturation or pressure, and the dependence of saturation on pressure. UNSAT is divided into three sections. The first section requires the user to specify the saturation-pressure (or capillary pressure) function. The second section requires the user to specify the derivative of saturation with respect to pressure. The third section requires the user to specify the relative permeability dependence on saturation or capillary pressure. INDAT2 requires only values of saturation, BASIS2 requires only values of saturation and relative permeability, NODALB and BUDGET require values of saturation and its pressure derivative. These calculations are controlled in UNSAT by the parameter IUNSAT which INDAT2 sets to a value of three, which BASIS2 sets to a value of two, and NODALB and BUDGET set to one. For simulation of purely saturated flow, IUNSAT is set to zero by INDAT1, and UNSAT is never called. The user may program these functions in any convenient way, for example, through data statements, calls to other programs, logical structures, 'read' or 'write' statements, or other preferred methods. More information may be found in section 7.5, "User-Supplied Programming."

Subroutine GLOBAN

-Purpose:

To assemble elementwise integrations into global matrix form.

-Called by:

ELEMEN

-Description:

GLOBAN carries out the sum over elements of integrals evaluated over each element by ELEMEN as suggested by relation (3.23). Both the matrix and right side vector terms involving integrals in the solution equations (4.65b) and (4.96b) are constructed.

Subroutine NODALB

-Purpose:

To calculate and assemble all nodewise and cellwise terms in the matrix equation.

-Called by:

SUTRA

-Calls to:

UNSAT

-Description:

NODALB undertakes a loop through all nodes in the mesh and calculates values of all cellwise terms. For each node, time derivatives and a fluid source are added to the fluid mass balance matrix equation. The time derivative as well as terms due to fluid sources production and boundary fluxes of U are prepared and added to the solute mass/energy balance matrix equation. Subroutine

UNSAT is called for unsaturated flow parameters. The terms added by NODALB may be described as the non-integral terms of (4.52) and (4.85) (except for the specified pressure terms.)

Subroutine BCB

-Purpose:

1) To implement specified pressure node conditions in the matrix equations.
2) To implement specified temperature or concentration node conditions in the matrix equations.

-Called by:

SUTRA

-Description:

The source terms involving ν_i in (4.52) are added to fluid balance matrix equation in order to obtain specified p nodes. The unified energy-solute mass balance is modified by the addition of a source, QPL, (calculated with the most recent p solution by subroutine SUTRA) with concentration or temperature value, UBC.

For a specified U node, the discretized balance equation is modified by zeroing the row of the U-matrix which gives the equation for the specified node. A one is placed on the diagonal and the specified U-value, UBC, is placed in the same row of the right side vector.

Subroutine PINCHB

-Purpose:

To implement pinch-node conditions in both matrix equations.

-Called by:

SUTRA

-Description:

PINCHB undertakes a loop through all pinch nodes. For each pinch node, the appropriate row of each matrix (for p and U) is zeroed, a one is placed on the diagonal, -0.5 is placed in the two columns corresponding to the side neighbors of the pinch node, and the corresponding element of the right side vector is zeroed.

Subroutine SOLVEB

-Purpose:

To solve a matrix equation with a non-symmetric banded matrix.

-Called by:

SUTRA

-Description:

SOLVEB expects the matrix band as a vertical rectangular block with the main diagonal in the center column, and minor diagonals in the other columns. The upper left-hand corner and lower right-hand corner of the matrix is blank.

The first section of the routine carries out an LU decomposition of the matrix which is saved within the original matrix space. The second section of the routine prepares the right side for solution and carries out back-substitution with a given right side vector.

Subroutine BUDGET

-Purpose:

1) To calculate and output a fluid mass budget on each time step with output.
2) To calculate and output a solute mass or energy budget on each time step with output.

-Called by:

SUTRA

-Calls to:

UNSAT, ADSORB

-Description:

BUDGET calculates and outputs a fluid mass, solute mass or energy budget on each output time step for whichever of p and/or U are solved for on the just-completed time step. The calculations are done as described in section 5.6 "Budget Calculations."

Subroutine STORE

-Purpose:

To store p and U results as well as other parameters on Unit-66 in a format ready for use as initial conditions in Unit-55. This acts as a backup for re-start in case a simulation is unexpectedly terminated before completion by computer malfunction.

-Called by:

SUTRA

-Description:

STORE is called upon completion of each time step of a simulation, if the storage option has been chosen. STORE writes the most recent solution for p and U at the nodes on a file, Unit-66, in a format exactly equivalent to that of input data file Unit-55. Information is also written which is used in a warm start (restart) of the simulation. The results of only the most recent time step are stored on UNIT-66 as STORE rewinds the file each time before writing.

SUTRA SIMULATION EXAMPLES

(p. 177 follows)

Chapter 6

Simulation Examples

This chapter outlines a number of example simulations which serve to demonstrate some of the capabilities of SUTRA modeling. Some of the examples show results which are compared with analytical solutions or numerical solutions available in the literature. These results serve to verify the accuracy of SUTRA algorithms for a broad range of flow and transport problems. The other examples demonstrate physical processes which SUTRA may simulate in systems where no other solutions are available. A complete SUTRA input data set and model output is provided for the example of section 6.3, "Radial Flow with Energy Transport," in Appendix B and Appendix C.

6.1 Pressure Solution for Radial Flow to a Well (Theis Analytical Solution)

Physical Set-up:

A confined, infinite aquifer contains a fully penetrating withdrawal well. Fluid is pumped out at a rate, Q_{TOT}.

Objective:

To simulate transient drawdown in this system which should match the Theis solution. The Theis solution (Lohman, 1979) is given in terms of variables used in SUTRA by:

$$s^* = \frac{Q_{TOT}\mu}{4 \pi \rho^2 \Delta z \, k \, |g|} \, W(u) \qquad (6.1a)$$

where s^* is the drawdown, W(u) is the well function of u, and

$$u = \frac{r^2 \mu S_{op}}{4 k t} \qquad (6.1b)$$

where r is the radial distance from the well to an observation point and t is the elapsed time since start of pumping.

Simulation Set-up:

The mesh contains one row of elements with element width expanding by a constant factor, 1.2915, with increasing distance from the well; other mesh dimensions are Δr_{min}=2.5 [m], Δr_{max}=25. [m], r_{max}=500. [m], Δz=1. [m]. Mesh thickness at node i, is given by $B_i=2\pi r_i$, which provides a radial coordinate system. The number of nodes and elements in the mesh are: NN=54, NE=26. See Figure 6.1.

The initial time step is, Δt_o=1. [s], with time steps increasing by a factor, 1.5, on each subsequent step.

One pressure solution is obtained per time step, solutions for concentration are ignored; the cycling parameters are: NPCYC=1, NUCYC=9999.

Parameters:

$S_{op} = 1.039 \times 10^{-6}$ [m·s^2/kg] $\qquad \varepsilon = 0.20$

$\alpha = 1.299 \times 10^{-6}$ [m·s^2/kg] $\qquad k = 2.0387 \times 10^{-10}$ [m^2]

$\beta = 4.4 \times 10^{-10}$ [m·s^2/kg] $\qquad \rho = 1000.$ [kg/m^3]

$|\underline{g}| = 9.81$ [m/s^2]

$Q_{TOT} = 0.6284$ [kg/s] (one-half at each well node)

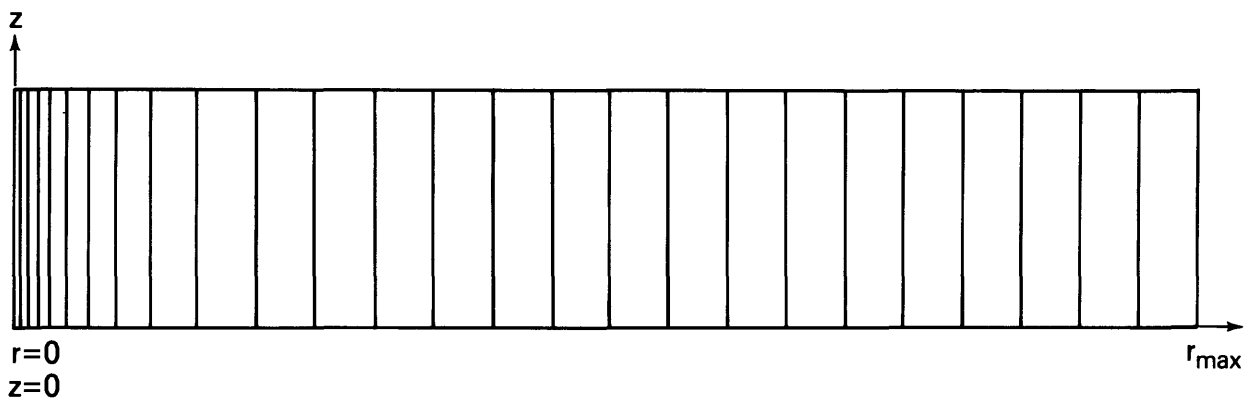


Figure 6.1
Radial finite-element mesh for Theis solution.

Boundary Conditions:

No flow occurs across any boundary except where hydrostatic pressure is specified at r_{max}. At the top outside corner of the mesh, r_{max}, pressure is held at zero. A sink is specified at r=0 to represent the well.

Initial Conditions:

Hydrostatic pressure with p=0 at the top of the aquifer is set initially.

Results:

SUTRA results are plotted for two locations in the mesh representing observation wells at r=15.2852 [m], and r=301.0867 [m]. Both locations should plot on the same Theis curve. The match of SUTRA results between one and 6000 minutes with the Theis analytical solution shown in Figure 6.2 is good.

6.2 Radial Flow with Solute Transport (Analytical Solutions)

Physical Set-up:

A confined infinite aquifer contains a fully penetrating injection well. Fluid is injected at a rate, Q_{TOT}, with a solute concentration, C^*, into the aquifer initially containing fluid with solute concentration, C_o. The fluid density does not vary with concentration.

Objective:

To simulate the transient propagation of the solute front as it moves radially away from the well. The concentrations should match the approximate

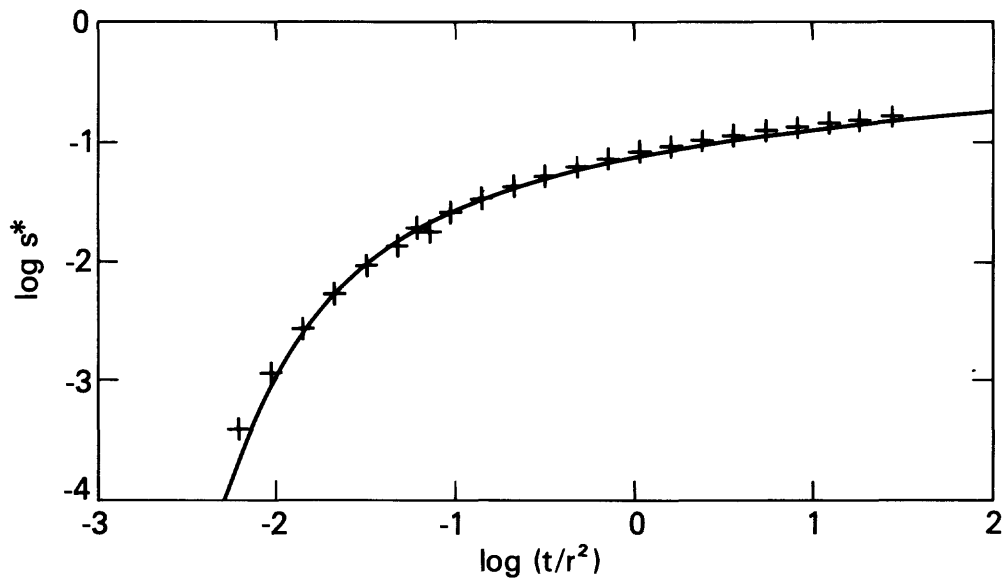


Figure 6.2
Match of Theis analytical solution (solid line) with SUTRA solution (+).

analytical solutions of Hoopes and Harleman (1967) and Gelhar and Collins (1971).

The solution of Gelhar and Collins (1971) is:

$$\left(\frac{C - C_o}{C^* - C_o}\right) = \frac{1}{2}\,\text{erfc}\left\{\frac{(r^2 - r^{*2})}{2[(\frac{4}{3}\,\alpha_L)r^{*3} + \left(\frac{D_m}{A}\right)r^{*4}]^{\frac{1}{2}}}\right\} \qquad (6.2)$$

where:

$$r^* = (2At)^{\frac{1}{2}} \qquad (6.3a)$$

$$A = \left(\frac{Q_{TOT}}{2\pi\epsilon b\rho}\right) \qquad (6.3b)$$

The Hoopes and Harleman (1967) solution is obtained by replacing r^* in the denominator of (6.2) with r.

Simulation Set-up:

The mesh consists of one row of elements with element width expanding from Δr_{min}=2.5 [m] by a factor, 1.06, to r=395. [m], and then maintaining constant element width of Δr=24.2 [m] to r_{max}=1000. [m]. Element height, b, is 10. [m]. Mesh thickness is set for radial coordinates, $B_i = 2\pi r_i$, with the number of nodes and elements given by NN=132, NE=65. See Figure 6.3.

The time step is constant at Δt=4021. [s], and outputs are obtained for times steps numbered: 225, 450, 900, 1800. One pressure solution is carried out to obtain a steady-state, (ISSFLO=1), and one concentration solution is done per time step, (NUCYC=1).

Parameters:

S_{op} = 0.0

$k = 1.02\text{x}10^{-11}$ [m^2]

ρ = 1000. [kg/m^3]

$D_m = 1.\text{x}10^{-10}$ [m^2/s]

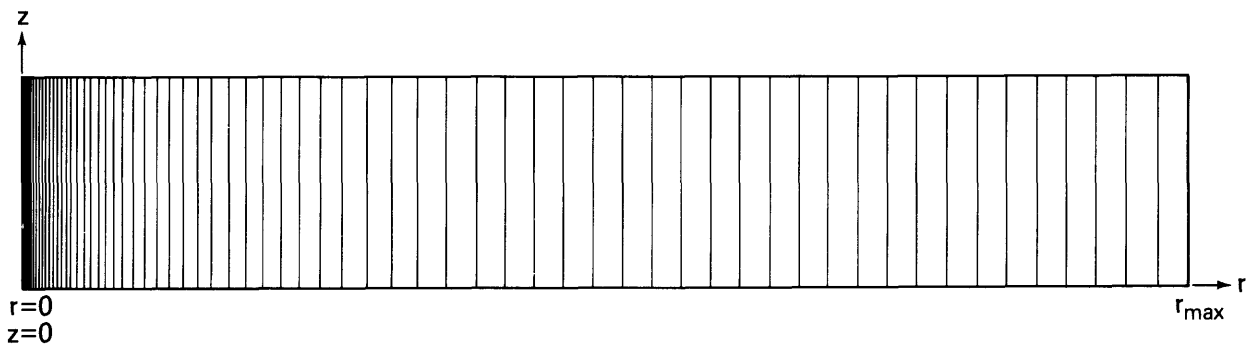


Figure 6.3
Radial finite-element mesh for constant-density solute and energy transport examples.

$\varepsilon = 0.2$ $\alpha_L = 10.0$ [m]

$\mu = 1.0 \times 10^{-3}$ [kg/m·s] $\alpha_T = 0.0$ [m]

$|\underline{g}| = 9.8$ [m/s^2]

$Q_{TOT} = 62.5$ [kg/s] (one half at each well node)

$C^* = 1.0$

Boundary Conditions:

No flow occurs across any boundary except where hydrostatic pressure is specified at r_{max}. At the top outside corner of the mesh, r_{max}, pressure is held at zero. A source is specified at r=0.0 to represent the injection well.

Initial Conditions:

Initially hydrostatic pressure is set with p=0.0 at the aquifer top. Initial concentration, C_o, is set to zero.

Results:

SUTRA results after 225, 450, 900 and 1800 time steps are compared with the approximate analytical solutions of Gelhar and Collins (1971) and Hoopes and Harleman (1967) in Figure 6.4. The analytical solutions are approximate and they bound the SUTRA solution at the top and bottom of the solute front. All solutions compare well with each other and the SUTRA solution may be considered to be more accurate than either approximate analytic solution because it is based on a very fine spatial and temporal discretization of the governing equation.

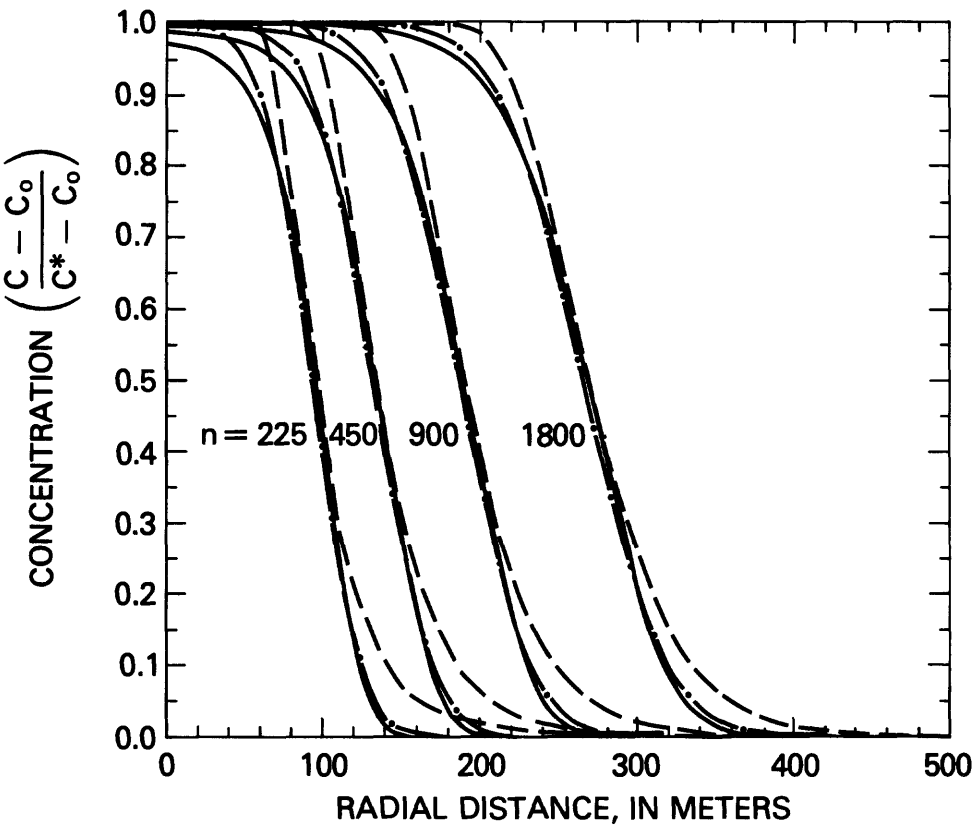


Figure 6.4
Match of analytical solutions for radial solute transport of Hoopes and Harleman (1967) (dashed), Gelhar and Collins (1971), (solid), and SUTRA solution (dash-dot). Number of elapsed time steps is n.

6.3 Radial Flow with Energy Transport

(Analytical Solution)

Physical Set-up:

A confined aquifer contains a fully penetrating injection well. Fluid is injected at a rate, Q_{TOT}, with a temperature, T^*, into the aquifer initially at a temperature, T_o. For this problem, density ρ, and viscosity μ, are kept approximately constant by injecting fluid that only slightly differs in temperature from the ambient fluid; i.e. (T^*-T_o) is small.

Objective:

To simulate the transient propagation of the temperature front as it radially moves away from the well. The solution should match an approximate analytical solution of Gelhar and Collins (1971) modified for energy transport. The Gelhar and Collins (1971) solution, as modified for energy transport is:

$$\left(\frac{T - T_o}{T^* - T_o}\right) = \frac{1}{2}\,\mathrm{erfc}\left\{\frac{(r^2 - r^{*2})}{2[(\frac{4}{3}\alpha_L)r^{*3} + \left(\frac{\lambda_{TOT}}{A_T}\right) r^{*4}]^{\frac{1}{2}}}\right\} \tag{6.4}$$

$$A = \frac{Q_{TOT}}{2\pi\epsilon B\rho} \tag{6.5}$$

$$A_T = \left(\frac{\epsilon\rho c_w}{c_{TOT}}\right) A \tag{6.6}$$

$$c_{TOT} = \epsilon\rho c_w + (1-\epsilon)\rho_s c_s \tag{6.7}$$

$$\lambda_{TOT} = \epsilon\lambda_w + (1-\epsilon)\lambda_s \tag{6.8}$$

$$r^* = (2A_T t)^{\frac{1}{2}} \tag{6.9}$$

The energy solution above may be obtained from the solute solution by retarding the velocity of transport to represent movement of an isotherm rather than a parcel of solute. This is done by accounting for energy storage in the solid grains of the aquifer material in the storage term of the analytical solution.

Simulation Set-up:

The mesh used for this example is the same as for the radial solute transport example. Time steps and frequency of SUTRA outputs are the same as for the radial solute transport example. Further, cycling of the SUTRA solution is the same as for the radial solute transport example.

Parameters:

c_w = 4182. [J/kg·°C]

c_s = 840. [J/kg·°C]

λ_w = 0.6 [J/s·m·°C]

ρ = 1000. [kg/m^3]

λ_s = 3.5 [J/s·m·°C]

ρ_s = 2650. [kg/m^3]

$\frac{\partial \rho}{\partial T}$ = 0.0

$\mu = \mu(T)$ (relation (2.5))

S_{op} = 0.

$k = 1.02 \times 10^{-11}$ [m^2]

ε = 0.2

$|g|$ = 9.8 [m/s^2]

α_L = 10. [m]

α_T = 0.0 [m]

Q_{TOT} = 312.5 [kg/s] (one half at each well node)

T^* = 1.0°C

Boundary Conditions:

No flow occurs across any boundary except where hydrostatic pressure is specified at r_{max}. At the top outside corner of the mesh, pressure is held at zero. A source is specified at r=0.0 to represent the injection well. Further, the system is thermally insulated along the top and bottom of the mesh.

Initial Conditions:

Initially, hydrostatic pressure is set with p=0.0 at the top of the aquifer. The initial temperature is T_o=0.0°C.

Results:

SUTRA results after 225, 450, 900 and 1800 time steps are compared with the approximate (modified) analytical solution of Gelhar and Collins (1971) in Figure 6.5. The analytic solution has the same relation to the SUTRA solution as it does in Figure 6.4 for solute transport. Thus the match is good, and again the SUTRA result may be more accurate than the approximate analytic result because of the fine discretization employed.

6.4 Areal Constant-Density Solute Transport (Example at Rocky Mountain Arsenal)

Physical Set-up:

This example involves a simple representation of ground-water flow and solute transport at the Rocky Mountain Arsenal, Denver, Colorado, which is based on the detailed model of the system by Konikow (1977). The simplified representa-

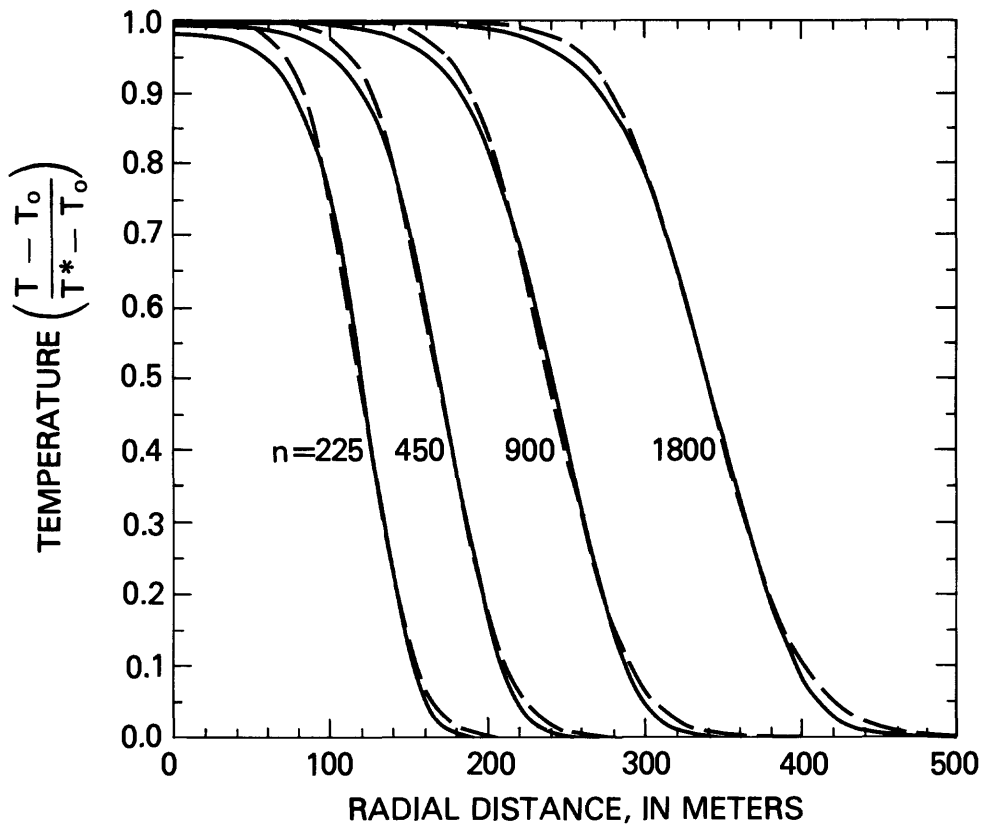


Figure 6.5
Match of analytical solution for radial energy transport (modified from Gelhar and Collins (1971) solid line) with SUTRA solution (dashed line). Number of elapsed time steps is n.

tion consists of an areal model of a rectangular alluvial aquifer with a constant transmissivity and two impermeable bedrock outcrops which influence groundwater flow. (See Figure 6.6.)

Regional flow is generally from the south-east to the north-west where some discharge occurs at the South Platte River. This is idealized as flow originating in a constant head region at the top of the rectangle in Figure 6.6, and discharging to the river at the bottom of the rectangle which also acts as a specified head region. Three wells pump from the aquifer (at a rate of Q_{OUT} each), and contamination enters the system through a leaking waste isolation pond (at a rate of Q_{IN}, with concentration, C^*). The natural background concentration of the contaminant is C_o.

Objectives:

1) To demonstrate the applicability of SUTRA to an areal constant density solute transport problem; 2) To convert SUTRA input data values so the pressure results represent heads, and the concentration results are in [ppm]; and 3) To simulate steady-state flow and hypothetical steady-state distributions of the contaminating solute, both as a conservative solute, and as a solute which undergoes first order decay, assuming that the contamination source in the idealized system is at a steady-state.

Simulation Set-up:

The rectangular mesh consists of 16 by 20 elements each of dimension 1000. ft by 1000. ft, as shown in Figure 6.6. (NN=357, NE=320). Mesh thickness, B, is the actual aquifer thickness, assumed constant for the idealized model.

One steady-state pressure solution is obtained (ISSFLO=1), and one concen-

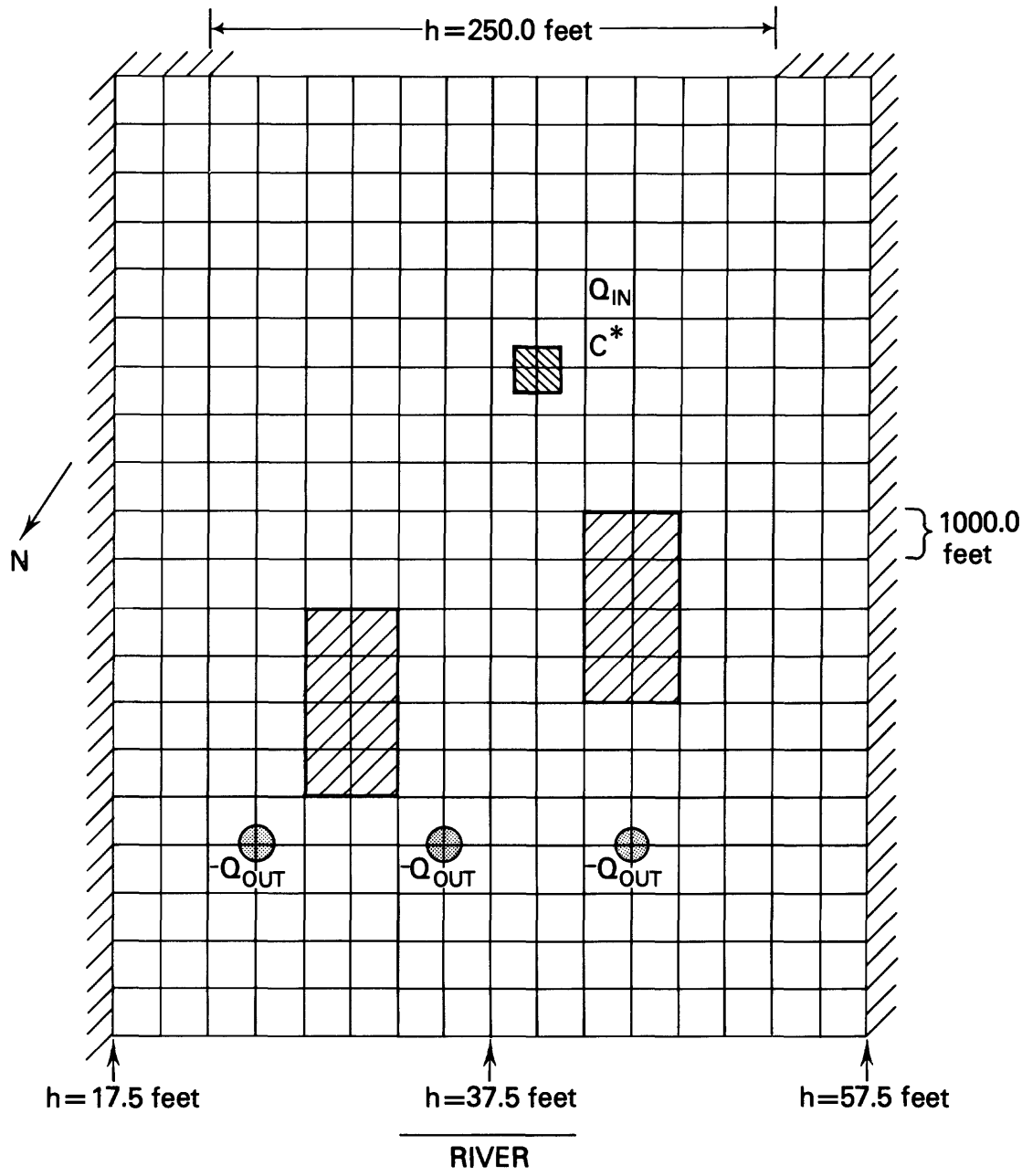


Figure 6.6
Idealized representation for example at Rocky Mountain Arsenal, and finite-element mesh.

tration solution is obtained. The concentration solution is obtained after a single time step of 1000. years, which, for all practical purposes, brings the concentration distribution to a steady-state.

The leaky pond is simulated as an injection of fluid (Q_{IN}, C^*) at a single node. Where the impermeable bedrock outcrop occurs, elements are assigned a conductivity value one-millionth of the aquifer values. A single value of constant head is specified along a portion of the top boundary, and a series of head values is specified along the bottom (river) boundary to represent changing elevation of the river.

In order to obtain results in terms of hydraulic head and [ppm], the following must be specified: ρ=1.0, $\frac{\partial \rho}{\partial C}$ = 0.0, $|\underline{g}|$=0.0, μ=1.0. Hydraulic conductivities are entered in the permeability input data set. Head values in [ft] are entered in data sets for pressure. Concentrations in [ppm] are entered in data sets for mass fraction concentration. Sources and sinks are entered in units of volume per time.

Parameters:

α_L = 500. [ft]

α_T = 100. [ft]

ϵ = 0.2

K = 2.5×10^{-4} [ft/s]
(hydraulic conductivity)

B = 40. ft

Q_{IN} = 1.0 [ft^3/s]

C^* = 1000. [ppm]

C_o = 10. [ppm]

Q_{OUT} = 0.2 [ft^3/s]
(at each of three wells)

Boundary Conditions:

No flow occurs across any boundary except where constant head is specified

at 250. [ft] at the top of the mesh and where constant head is specified as changing linearly between 17.5 [ft] at the bottom left corner, and 57.5 [ft] at the bottom right corner of the mesh. Inflow at the top of the mesh is at background concentration, C_o=10. [ppm]. A source is specified at the leaky pond node, and a sink is specified at each well node.

Initial Conditions:

Initial pressures are arbitrary for steady-state simulation of pressure. Initial concentration is C_o=10. [ppm].

Results:

A nearly steady-state solute plume for a conservative solute is obtained after a 1000 year time step shown in Figure 6.7. For a solute which undergoes first order decay with decay coefficient, γ=1.1x10^{-9} [s^{-1}] (approximately a 20 year half-life), the nearly steady plume is shown in Figure 6.8. Just upstream of the plume envelope is a region in which concentration dips slightly below background levels. This is due to a numerical problem of insufficient spatial discretization in a region where the concentration must change sharply from fresh upstream values to contaminated plume values. Lower dispersivity values would exacerbate the problem in the upstream region, but minor upstream oscillations do not affect concentration values within the plume.

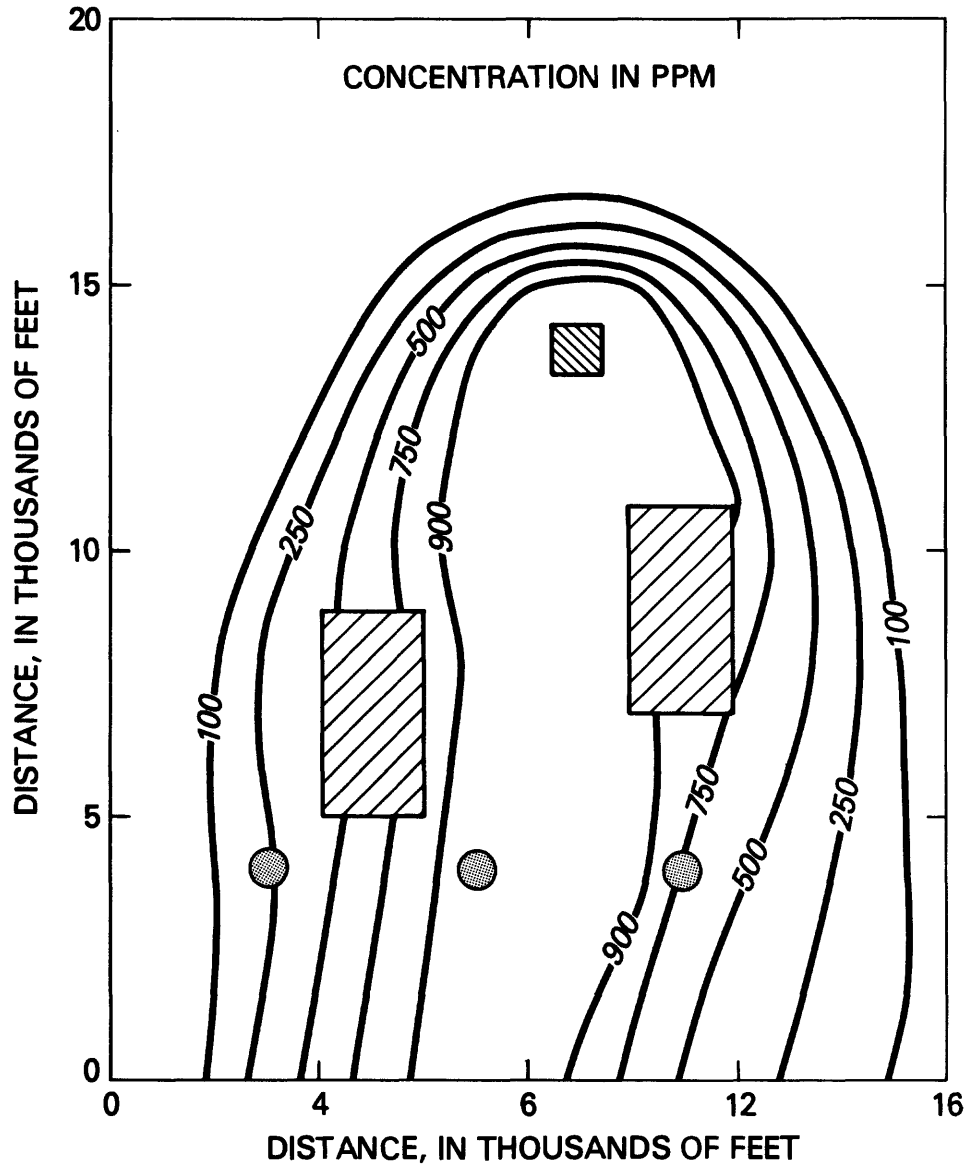


Figure 6.7
Nearly steady-state conservative solute plume as simulated for the Rocky Mountain Arsenal example by SUTRA.

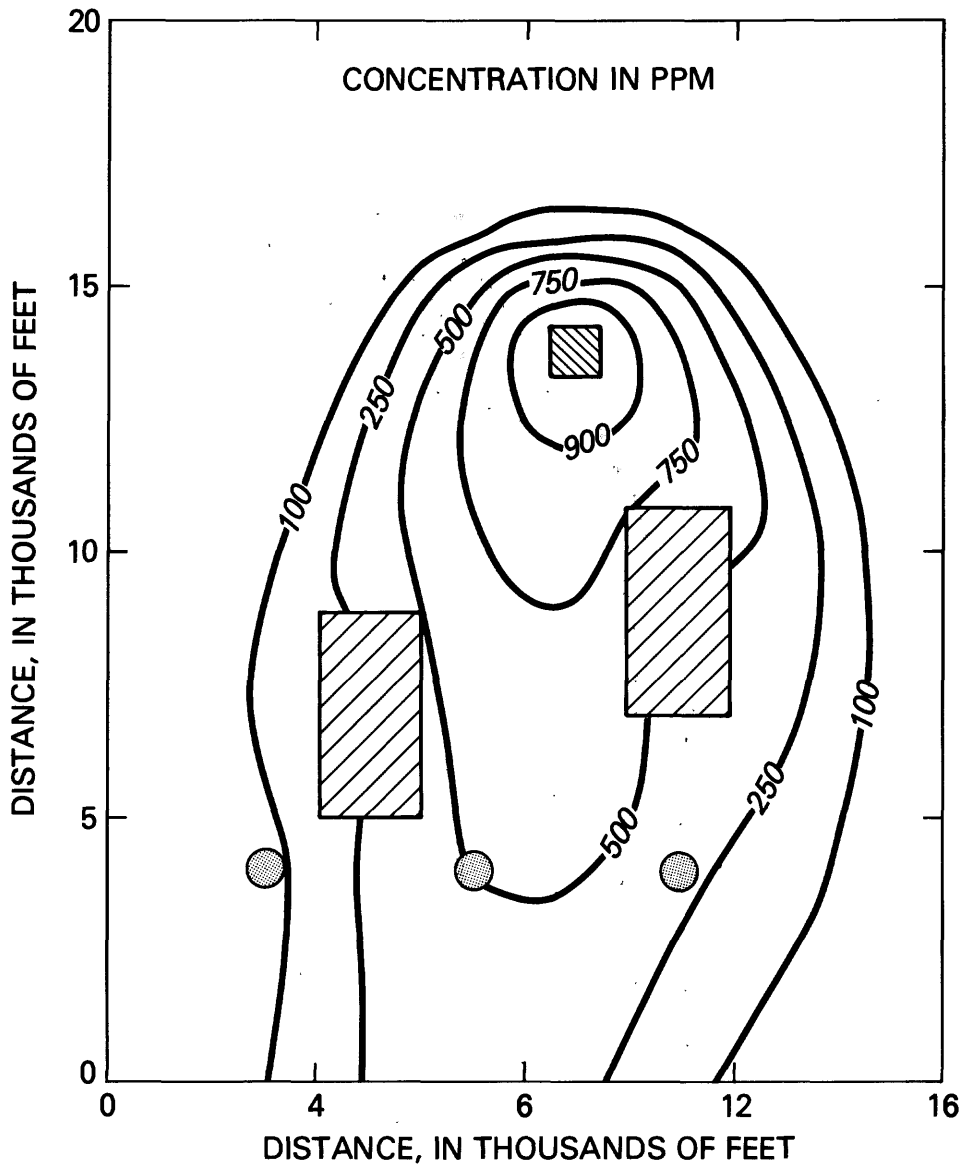


Figure 6.8
Nearly steady-state solute plume (with solute half-life ~ 20. years) as simulated for the Rocky Mountain Arsenal example by SUTRA.

6.5 Density-Dependent Flow and Solute Transport

(Henry (1964) Solution for Sea-Water Intrusion)

Physical Set-up:

This problem involves sea-water intrusion into a confined aquifer studied in cross-section under steady conditions. Fresh-water recharge inland flows over salt water in the section and discharges at a vertical sea boundary.

The intrusion problem is non-linear and may be solved by approaching the steady state gradually with a series of time steps. Initially there is no salt water in the aquifer, and at time zero, salt water begins to intrude the fresh water system by moving under the fresh water from the sea boundary. The intrusion is caused by the greater density of the salt water.

Dimensions of the problem are selected to make for simple comparison with the steady-state dimensionless solution of Henry (1964), and with a number of other published simulation models. A total simulation time of t=100. [min], is selected, which is sufficient time for the problem to essentially reach steady state at the scale simulated.

Simulation Set-up:

The mesh consists of twenty by ten elements, each of size 0.1 [m] by 0.1 [m], (NN=231, NE=200). Mesh thickness, B, is 1. [m]. See Figure 6.9. Time steps are of length 1. [min], and 100 time steps are taken in the simulation. Both pressure and concentration are solved for on each time step, (NUCYC=NPCYC=1).

A source of fresh water is implemented by employing source nodes at the left vertical boundary which inject fresh water at rate, Q_{IN}, and concentration, C_{IN}. The right vertical boundary is held at hydrostatic pressure of

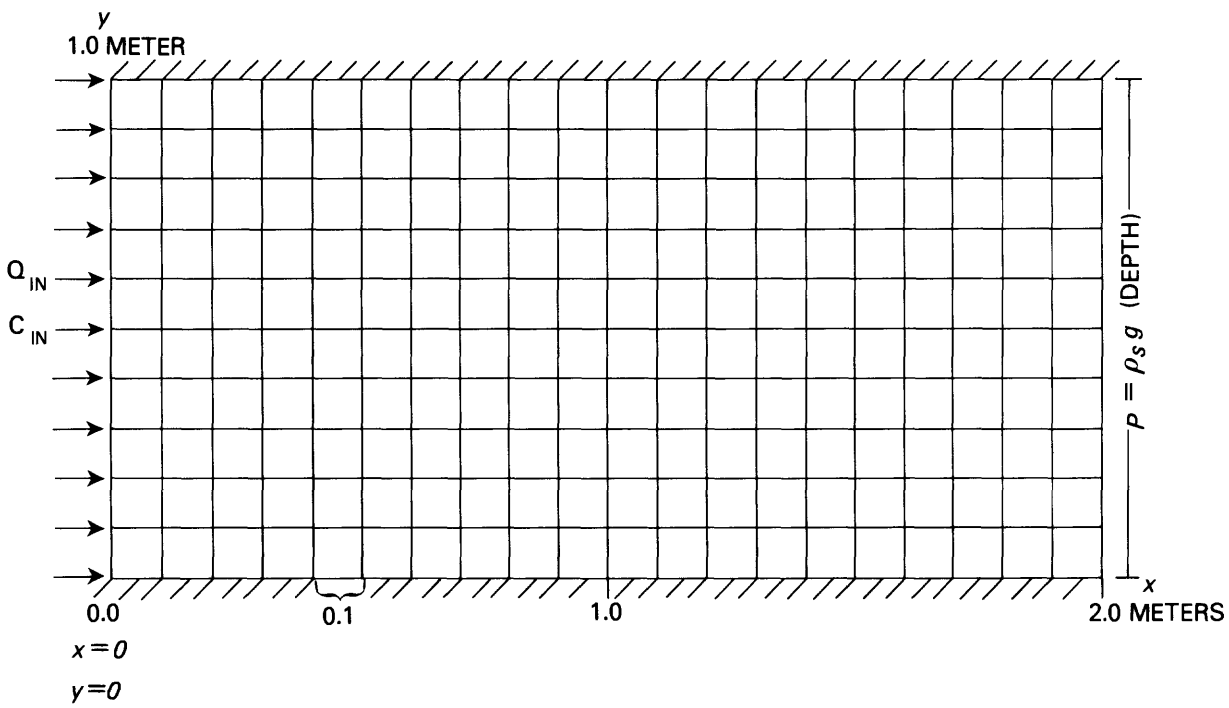


Figure 6.9
Boundary conditions and finite-element mesh for Henry (1964) solution.

sea water through use of specified pressure nodes. Any water which enters the section through these nodes has concentration C_{BC} of sea water (equal to C_s).

Parameters:

$\varepsilon = 0.35$

$C_s = 0.0357 \left[\frac{\text{kg(dissolved solids)}}{\text{kg(seawater)}}\right]$

$\rho_s = 1025.\ [\text{kg/m}^3]$

$\frac{\partial p}{\partial C} = 700. \left[\frac{\text{kg(seawater)}^2}{(\text{kg dissolved solids}\cdot\text{m}^3)}\right]$

$\rho_o = 700.\ [\text{kg/m}^3]$

$Q_{IN} = 6.6\text{x}10^{-2}$ [kg/s]
(divided among 11 nodes
at left boundary)

$k = 1.020408\text{x}10^{-9}\ [\text{m}^2]$
(based on $K=1.0\text{x}10^{-2}$ [m/s])

$|g| = 9.8\ [\text{m/s}^2]$

$\alpha_L = \alpha_T = 0.0$

$B = 1.0$ [m]

$D = \begin{cases} 6.6\text{x}10^{-6}\ [\text{m}^2/\text{s}] \\ 18.8571\text{x}10^{-6}\ [\text{m}^2/\text{s}] \end{cases}$ two cases

$C_{IN} = 0.0$

Boundary Conditions:

No flow occurs across the top and bottom boundaries. A fresh-water source is set along the left vertical boundary. Specified pressure is set at hydrostatic sea water pressure with (ρ_s=1025. [kg/m^3]) along the right vertical boundary. Any inflowing fluid at this boundary has the concentration, C_s=0.0357 [kg(dissolved solids)/kg(seawater)], of sea water.

Initial Conditions:

Natural steady pressures are set everywhere in the aquifer based on the fresh-water inflow, zero concentration everywhere, and the specified pressures at the sea boundary. These initial conditions are obtained through an extra initial simulation which calculates steady pressures under these conditions.

Results:

Henry's solution assumes that dispersion is represented by a constant large coefficient of diffusion, rather than by velocity-dependent dispersivity. Two different values of this diffusivity have apparently been used in the literature by those testing simulators against Henry's solution. The total dispersion coefficient of Henry (1964), D, is equivalent to the product of porosity and molecular diffusivity in SUTRA, $D=\varepsilon D_m$.

Henry's results are given for his non-dimensional parameters: $\xi=2.0$, $b=0.1$, $a\simeq.264$ (page C80- Figure 34 in Henry (1964)). In order to match the Henry parameters using simulation parameters as listed above, values of $D=6.6\times10^{-6}$ [m^2/s] and $D_m=18.8571\times10^{-6}$ [m^2/s] are required. Some authors, however, have apparently used a value equivalent to $D_m=6.6\times10^{-6}$ [m^2/s] and $D = 2.31\times10^{-6}$ [m^2/s], which differs from the Henry parameters by a factor equal to the porosity.

In the previous model solutions compared here, only Huyakorn and Taylor (1976) have employed the higher value which should match Henry's solution. A comparison of SUTRA results at t=100. [min], using the higher value with those of Huyakorn and Taylor (1976) along the bottom of the section is shown in Figure 6.10. Huyakorn and Taylor's results are for a number of simulation models based on significantly different numerical methods. SUTRA results are also shown for the lower diffusivity value. The results of simulations using the higher diffusivity value compare favorably. Results using the higher value have also been obtained with the INTERA (1979) finite-difference code at t=100. [min], (with centered-in-space and centered-in-time approximations). These are compared with SUTRA and the Henry solution for the 0.5 isochlor in Figure 6.11. The models match well but do not compare favorably with the analytic

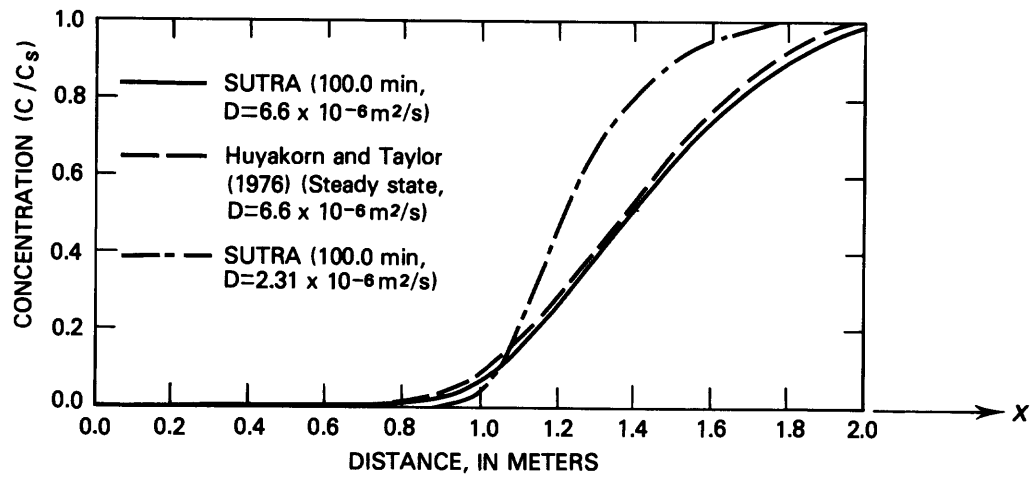


Figure 6.10
Match of isochlors along bottom of aquifer for numerical results of Huyakorn and Taylor (1976) and SUTRA.

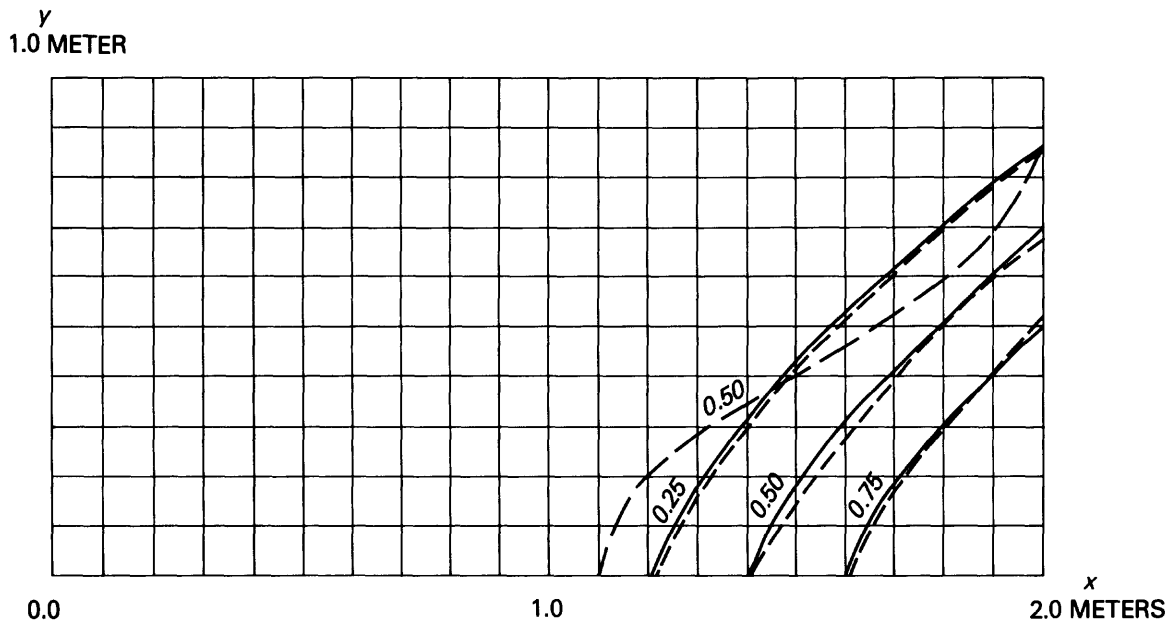


Figure 6.11
Match of isochlor contours for Henry analytical solution (for 0.50 isochlor) (long dashes), INTERA code solution (short dashes), SUTRA solution (solid line).

solution, which is approximate and may not be as accurate as the numerical solutions.

For the lower value of diffusivity, D_m=6.6x10^{-6} [m^2/s], (which should not compare with the Henry result), the SUTRA solution at t=100. [min] is compared in Figure 6.12 with that of Pinder and Cooper (1970) (method of characteristics), Segol et. al. (1975) (finite elements), Desai and Contractor (1977) (finite elements - coarse mesh), and Frind (1982) (finite elements). The match of the numerical 0.5 isochlor solutions is remarkably good; however, it should be noted that none of these match the analytical solution.

6.6 Density-Dependent Radial Flow and Energy Transport (Aquifer Thermal Energy Storage Example)

Physical Set-up:

This is an example of aquifer thermal energy storage. Hot water is injected into an aquifer for storage and later withdrawn and used as an energy source. The fully penetrating injection wells are emplaced in a well-field in a hexagonal packing pattern. The wells are at the vertices of contiguous equilateral triangles with sides of 500. [m]. This gives approximately radial symmetry to physical processes surrounding an interior well.

Objective:

To simulate the initial injection-withdrawal cycle at an interior well consisting of 90 days of injection (at Q_{IN}) of 60°C water into the aquifer initially at 20°C, and 90 days of withdrawal (at Q_{IN}) producing the stored water. Degradation of recovered fluid temperature should occur due to thermal

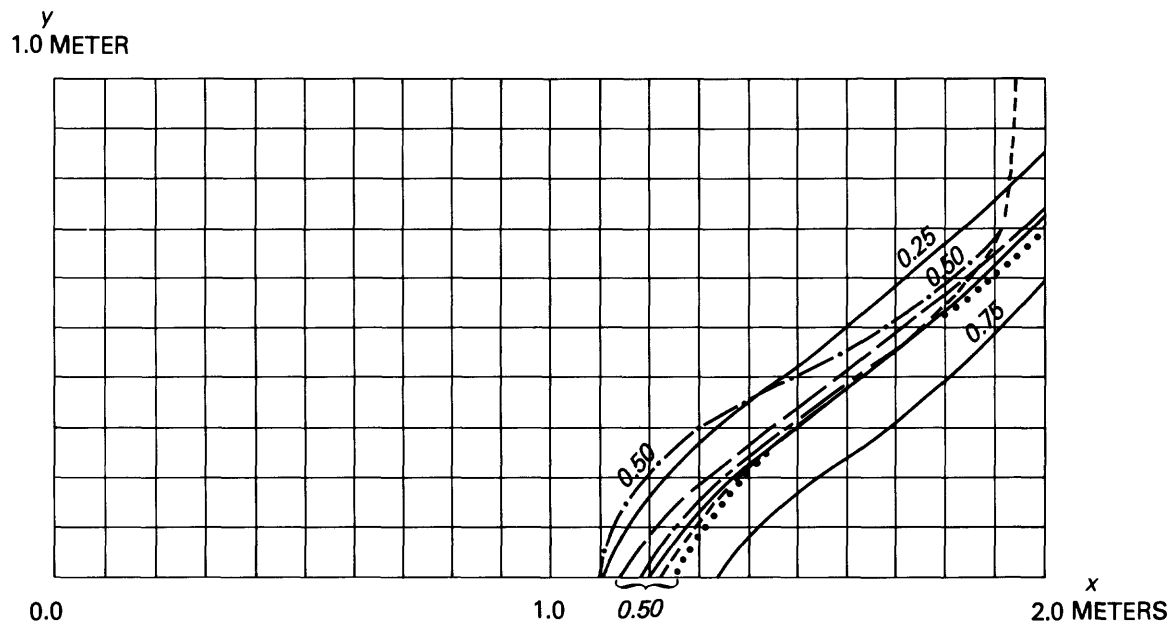


Figure 6.12
Match of 0.50 isochlor contours for Henry problem with simulated results for $D_m = 6.6 \times 10^{-9}[m^2/s]$ of Pinder and Cooper (1970), (short dashes), Segol, et al (1975) (dotted line), Frind (1982) (long and short dashes), Desai and Contractor (1977) (long dashes). SUTRA results at isochlors (0.25,0.50,0.75) (solid line). Henry (1964) solution for $D_m = 18.8571 \times 10^{-9}$ $[m^2/s]$, (0.50 isochlor, dash-dot).

conduction, dispersion, and tipping of the thermal front. The front should tip as less dense, less viscous hot water rises over colder, denser, and more viscous formation water.

Simulation Set-up:

The mesh is 30. [m] high with a vertical spacing between nodes of 3.0 [m]. The first column of elements has width Δr_{min} = 1.0 [m], and element width increases with each column by a factor, 1.1593, to a final column of width, Δr_{max} = 35. [m]. The outside boundary of the mesh is at r_{max} = 246. [m]. See Figure 6.13. Mesh thickness, B, at any node i, is B_i = $2\pi r_i$, giving cylindrical symmetry. The number of nodes and elements in the mesh is given by NN=286, NE=250.

The time step is constant at Δt = 3.0 [days]. One pressure solution and one temperature solution is obtained at each time step (NPCYC=NUCYC=1). The storage coefficient is assumed negligible resulting in a steady flow field at any time step. Subroutine BCTIME is programmed to control the well rate which changes after 90 days from fluid injection to fluid withdrawal.

A time-dependent fluid source is specified at the left vertical boundary (center axis) which injects 60.[°C] water for 90 days and then withdraws ambient water for 90 days. The right vertical boundary is held at hydrostatic pressure for water at 20. [°C]. Any inflow at this boundary has a temperature of 20.°C. Thermally insulated and impermeable conditions are held at the top and bottom of the mesh.

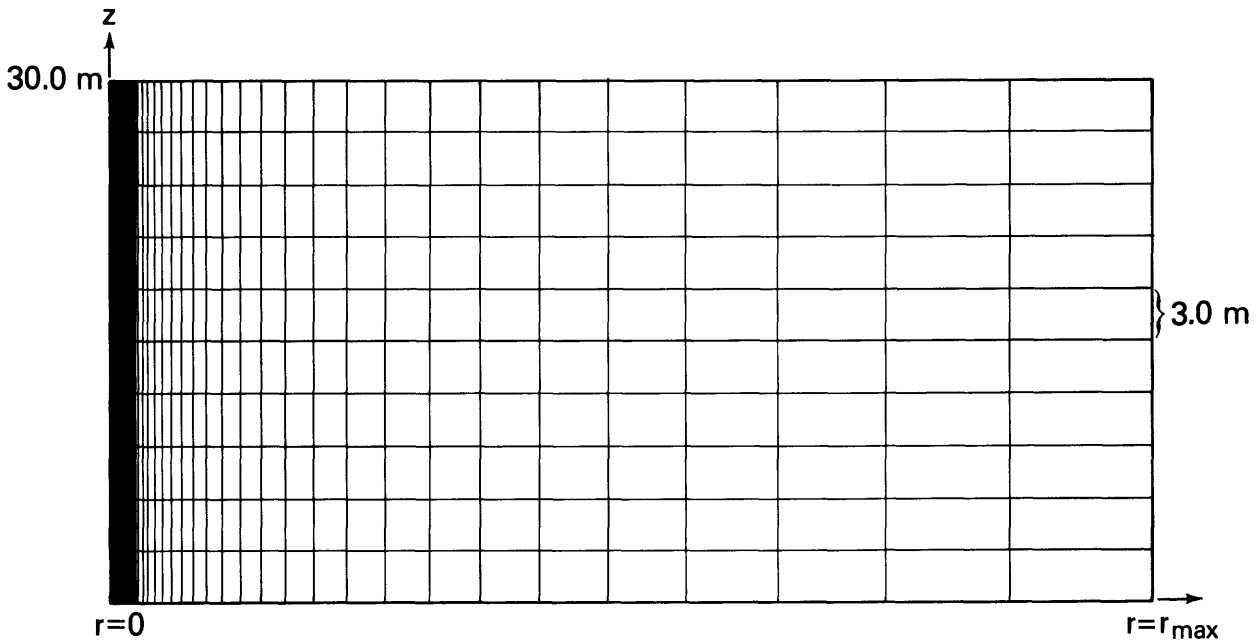


Figure 6.13
Radial two-dimensional finite-element mesh for aquifer thermal energy storage example.

Parameters

c_w = 4182. [J/kg·°C]

c_s = 840. [J/kg·°C]

λ_w = 0.6 [J/s·m·°C]

λ_s = 3.5 [J/s·m·°C]

T_o = 20.°C

$\frac{\partial \rho}{\partial T}$ = -0.375 [kg/m^3·°C]

T* = 60.[°C]

Q_{TOT} = 200. [kg/s]
(distributed along well)

S_{op} = 0

k = 1.02 x 10^{-10} [m^2]

ε = 0.35

ρ_o = 1000. [kg/m^3]

ρ_s = 2650. [kg/m^3]

μ = μ(T) (relation (2.5))

|g| = 9.81 [m/s^2]

α_L = 4.0 [m]

α_T = 1.0 [m]

Boundary Conditions:

Conditions of no flow and thermal insulation are held at all boundaries except where hydrostatic pressure at T = 20.[°C] is specified at r_{max}. At the top outside corner of the mesh the pressure is held at zero. A time-dependent source is specified at r=0.0 to represent the injection-withdrawal well.

Initial Conditions:

Hydrostatic pressure is specified initially, with p=0.0 at the top of the aquifer. The initial temperature is set to T_o=20.[°C].

Results:

SUTRA results during injection after 30 days and 90 days are shown in Figure 6.14 and Figure 6.15. Simulated results during withdrawal are shown in

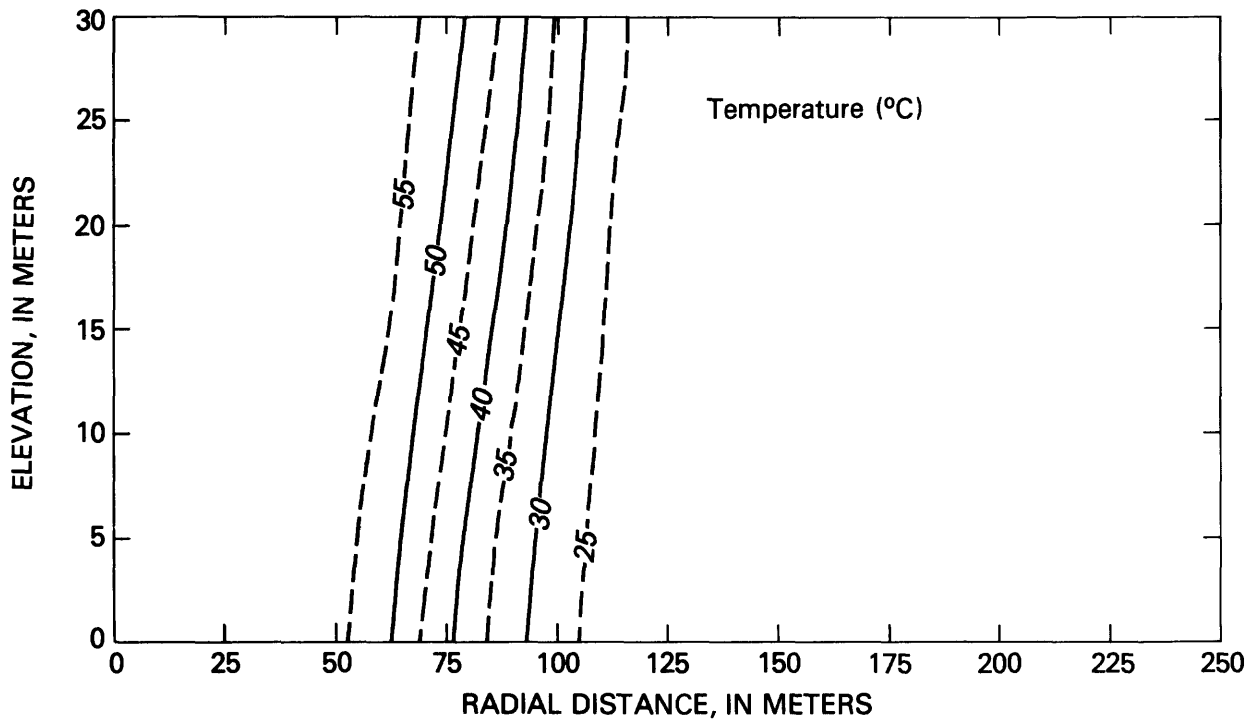


Figure 6.14
SUTRA results after 30 days of hot water injection.

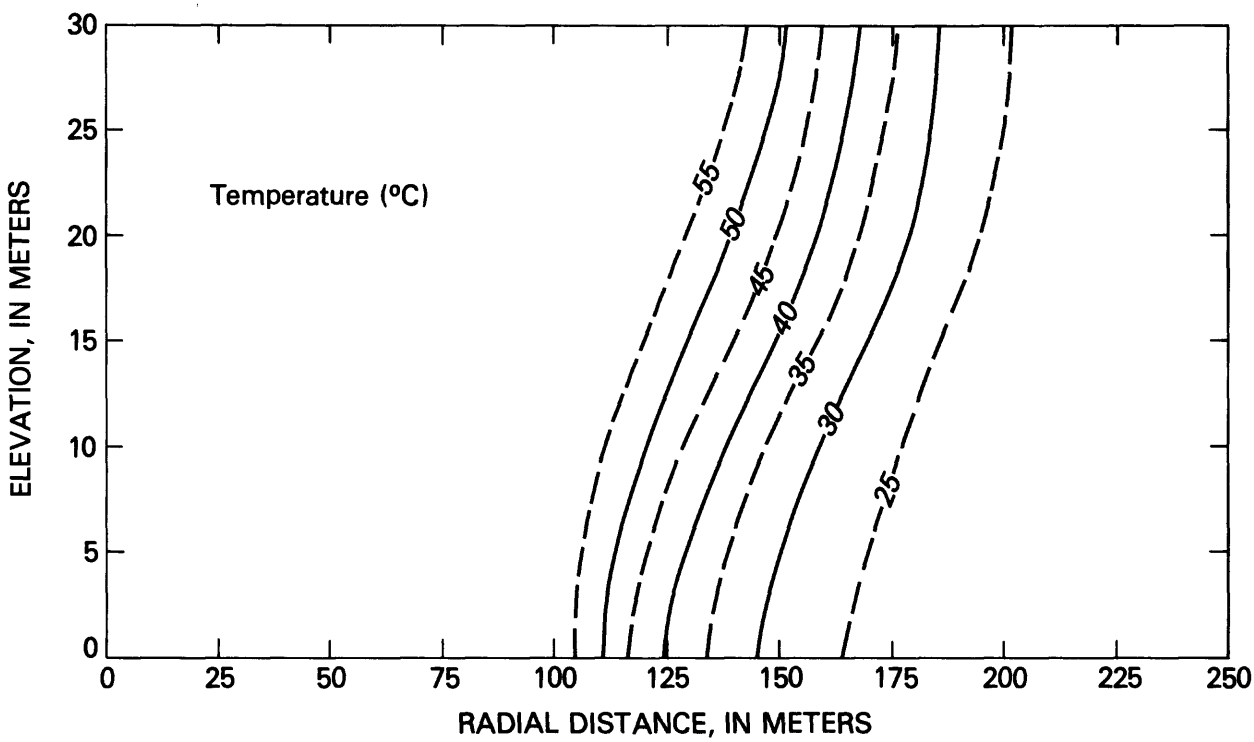


Figure 6.15
SUTRA results after 90 days of hot water injection.

Figure 6.16, Figure 6.17, and Figure 6.18 after 30 days, 60 days, and 90 days of withdrawal. The thermal transition zone (between hot and cold water) widens throughout the injection-production cycle, due to both dispersion and heat conduction. The top of the transition zone tips away from the well during the entire cycle, due to the bouyancy of the hotter water. These two effects combine to cause cooler water to reach the bottom of the withdrawal well much earlier than if no density differences or dispersion existed. Also, although the same quantity of water has been removed as injected, energy is lost to the aquifer during the cycle as seen at the end of simulation.

6.7 Constant-Density Unsaturated Flow and Solute Transport (Example from Warrick, Biggar and Nielsen (1971))

Physical Set-up:

Water containing solute infiltrates an initially unsaturated solute-free soil for about two hours. Solute-free water continues to infiltrate the soil after the initial two hours. The moisture front and a slug of solute move downwards through the soil column under conservative non-reactive constant-density transport conditions, as described in a field experiment by Warrick, Biggar, and Nielsen (1971).

Objective

To simulate the transient propagation of the moisture front and solute slug as they move downwards through the soil column, under conditions of simulation equivalent to that used by Van Genuchten (1982) to represent the

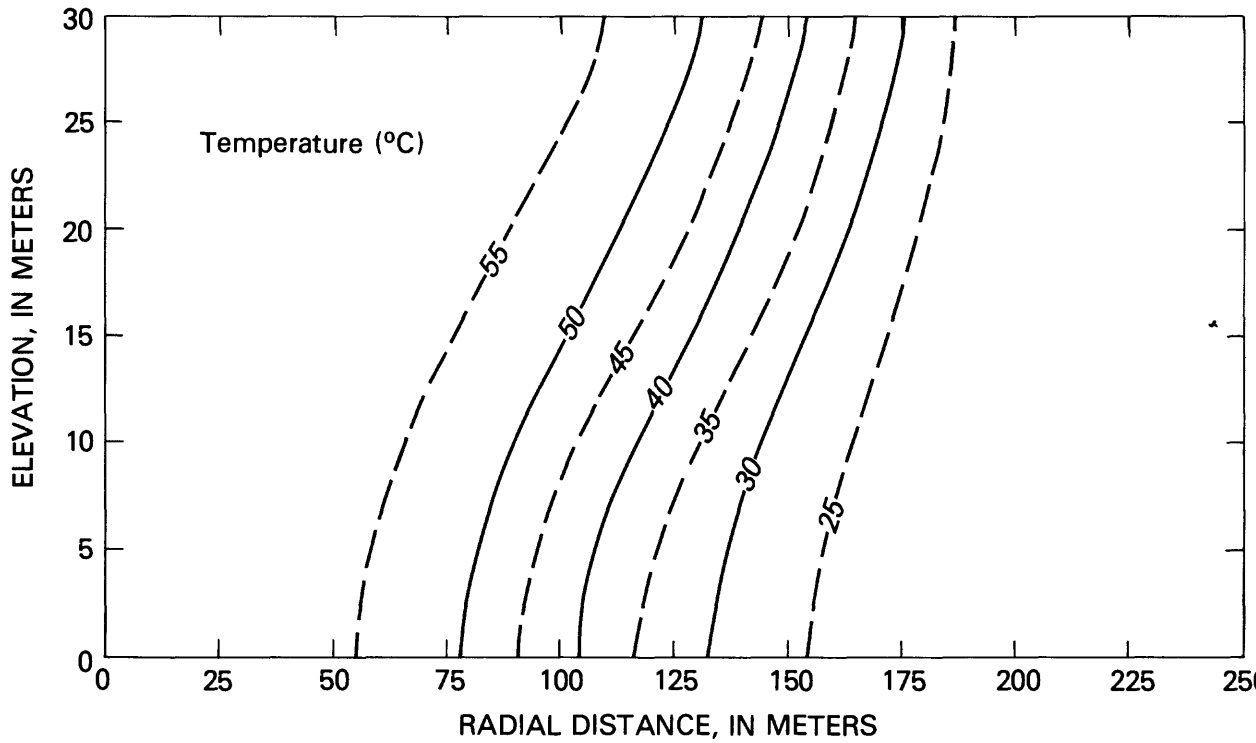


Figure 6.16
SUTRA results after 30 days of pumping, (120 days total elapsed time).

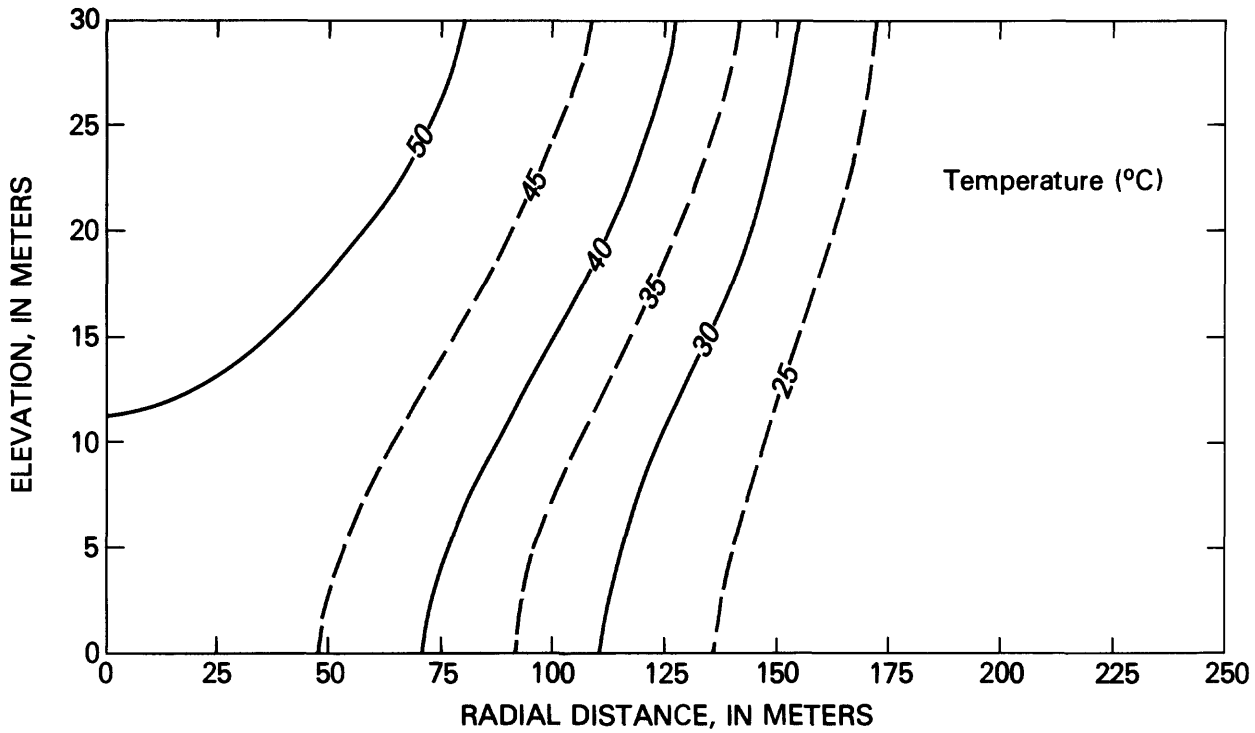


Figure 6.17
SUTRA results after 60 days of pumping, (150 days total elapsed time.)

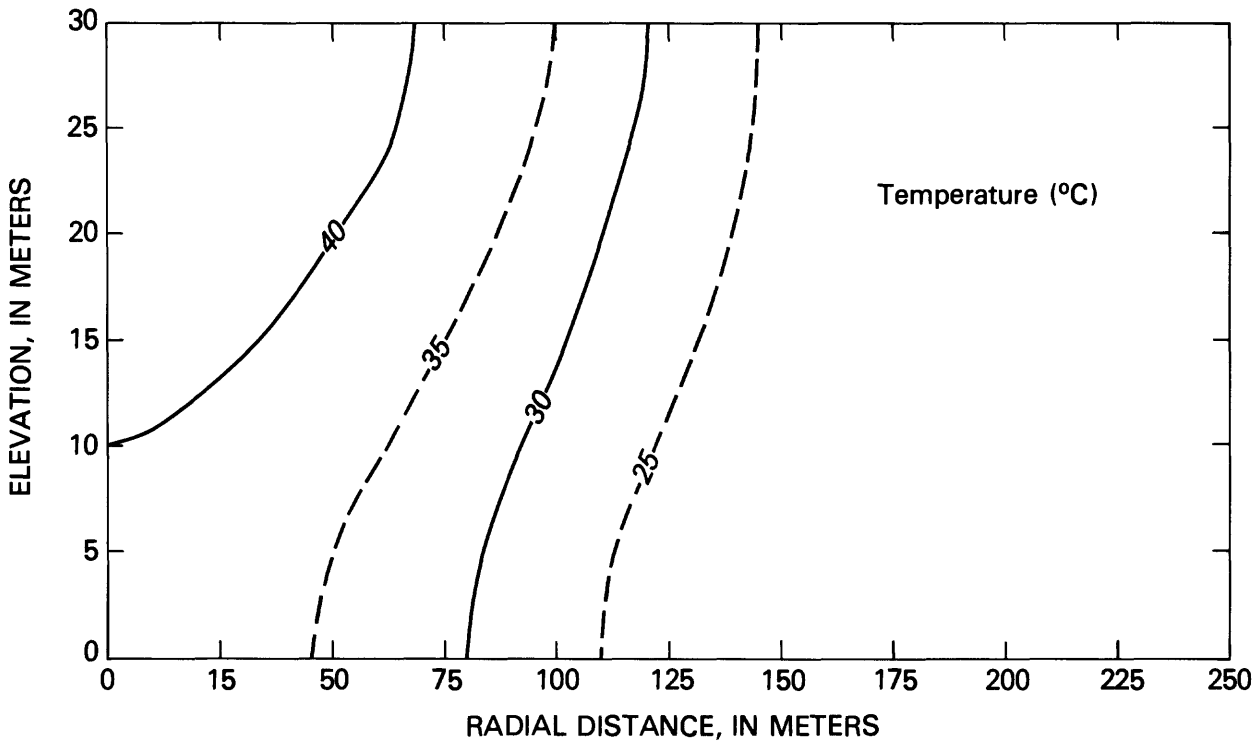


Figure 6.18
SUTRA results after 90 days of pumping, (180 days total elapsed time.)

field experiment. The solutions should match the best fine grid - fine time step simulation results of Van Genuchten (1982) which were obtained with a number of different finite difference and finite element numerical methodologies.

Simulation Set-up:

The mesh consists of a single vertical column of 100 elements oriented in the direction of gravity, which is 2.0 [m] long and 0.01 [m] wide. The number of nodes and elements is: NN = 202, NE = 100. Each element is 0.01 [m] wide and 0.02 [m] high. Mesh thickness is unity. The vertical coordinate, x, is measured downward from the top of the column.

The time step is constant at $\Delta t = 30.$ [s], and because of the small time step, only one iteration is done per step. The simulation is carried out for nine hours of infiltration.

Outputs are obtained once each hour, but are only compared at two hours and nine hours. There is one pressure solution and one concentration solution each time step.

Parameters

$$k_r = 1.235376 \times 10^{-6} \exp(13.604\, S_w) \tag{6.10}$$

$$\begin{cases} S_w = 1.52208 - 0.0718947 \ln(-p) & (6.11a) \\ \quad \text{for } -2892.38 < p \leq -1421.96 \ [\text{kg}/(\text{m}\cdot\text{s}^2)] \\ S_w = 2.94650 - 0.250632 \ln(-p) & (6.11b) \\ \quad \text{for } p < -2892.38 \ [\text{kg}/(\text{m}\cdot\text{s}^2)] \end{cases}$$

$S_{op} = 0.0$ $\qquad$ $\rho = 1000.$ [kg/m^3]

$k = 4.4558 \times 10^{-13}$ [m^2] $\qquad$ $\sigma_w = 0.0$

$\varepsilon = 0.38$ $\alpha_L = 0.01$ [m]

$\mu = 1.0 \times 10^{-3}$ [kg/m·s] $\alpha_T = 0.0$ [m]

$|\underline{g}| = 9.81$ [m/s^2]

Boundary Conditions

The top boundary representing an infiltration pond, is held fully saturated, $S_w = 1.0$, (water content $\varepsilon S_w = 0.38$) during the simulation by specification of pressure at $p = -1421.96$ [kg/(m·s^2)]. The bottom boundary is held at a specified saturation of $S_w = 0.526316$, (water content $\varepsilon S_w = 0.20$) by specification of pressure, $p = -15616.5$ [kg/(m·s^2)]. No flow occurs across either side boundary, but flow enters the top boundary due to the pressure specification. The concentration of inflowing fluid at the top is held at $C = 209.$ [meq/liter] until time $t = 168.0$ [min], at which time the concentration of the inflow drops to $C = 0.0$ [meq/liter]. Note that the concentration units are arbitrary (need not be mass fractions) because this is a constant density simulation.

Initial Conditions:

Initially, pressures are set to obtain the following initial distribution of saturation, shown in Figure 6.19:

$$S_w(x,t=0) = \begin{cases} 0.394737 + 0.219289\,x & 0.0 < x \leq 0.60 \text{ [m]} \\ 0.526316 & 0.6 < x \leq 1.25 \text{ [m]} \end{cases} \qquad (6.12)$$

Initial concentrations are set to zero.

Results:

SUTRA results after two hours and nine hours of infiltration are shown with the finely discretized solutions of Van Genuchten (1982) for saturation in Figure 6.19, and for concentration in Figure 6.20. The results coincide almost exactly for both early and late time so only one curve can be shown for each time. Although the SUTRA results are obtained with a non-iterative solution and small time steps, similar results may be obtained with longer time steps and a few iterations per step. The concentration front lags behind the moisture front as the volume between the concentration front and top boundary represents the water which has infiltrated. The volume of water between the moisture front and concentration front represents the initial water in the medium which has been displaced by the infiltrating water.

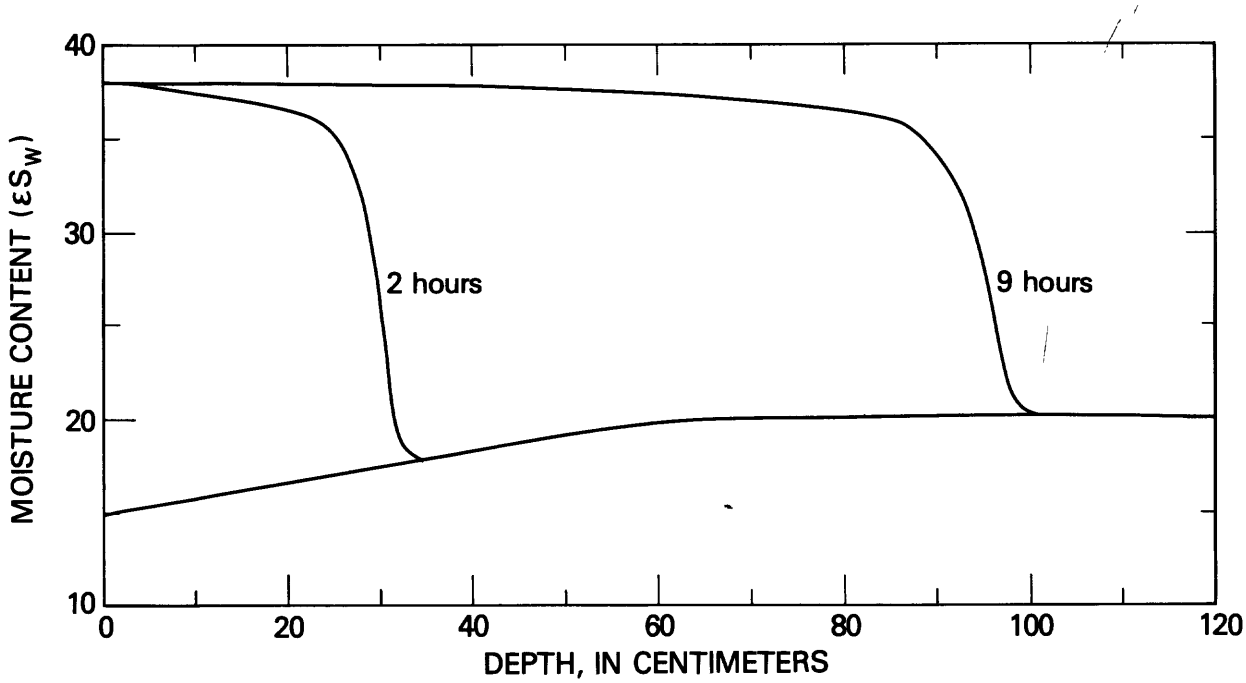


Figure 6.19
Propagation of moisture front for unsaturated flow and solute transport example. Results of Van Genuchten (1982) and SUTRA shown in same solid line.

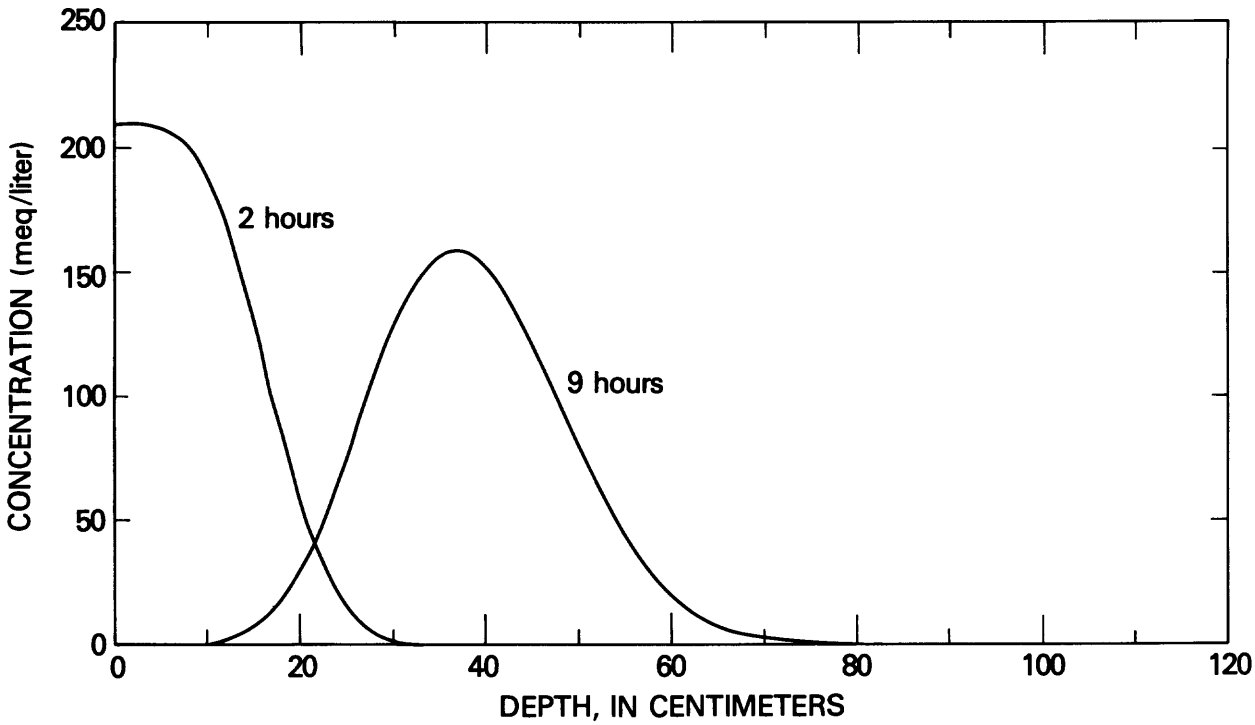


Figure 6.20
Propagation of solute slug for unsaturated flow and solute transport example. Results of Van Genuchten (1982) and SUTRA shown in same solid line.

(p.219 follows)

SUTRA SIMULATION SETUP

(p. 221 follows)

Chapter 7

Simulation Setup

7.1 SUTRA Data Requirements

The following is a complete list of data required to setup a simulation with SUTRA. (1) The information included in the list is the parameter name used in this report (if it has been mentioned), (2) the parameter units, (3) the parameter name in the input data list, and (4) a short explanation of the parameter.

Mesh and coordinate data

g_x	$[L/s^2]$	GRAVX	x-component of gravity vector
g_y	$[L/s^2]$	GRAVY	y-component of gravity vector
x_i	[L]	X(I)	x coordinate of node i, for all nodes in mesh
y_i	[L]	Y(I)	y coordinate of node i, for all nodes in mesh
NN		NN	total number of nodes in mesh
		IIN(1-4)	counter-clockwise nodal incidence list in each element
		IEDGE(1-4)	ordered list of pinch nodes in each element according to Figure 5.5
NE		NE	total number of elements in mesh
		NPINCH	total number of pinch nodes in mesh
		NBI	full band-width of global banded matrix

Flow parameters

β	$[M/(L\cdot s^2)]^{-1}$	COMPFL	fluid compressibility
α	$[M/(L\cdot s^2)]^{-1}$	COMPMA	solid matrix compressibility
ε_i	[1]	POR(I)	volumetric porosity of solid matrix at each node
k_{max_L}	$[L^2]$	PMAX(L)	maximum component of permeability in each element
k_{min_L}	$[L^2]$	PMIN(L)	minimum component of permeability in each element
θ_L	[°]	ANGLEX(L)	angle between k_{max} and +x-axis in each element
ρ_o	$[M/L^3]$	RHOWO	fluid base density
$\frac{\partial \rho}{\partial U}$	$[M/L^3\cdot °C]$	DRWDU	for energy transport: coefficient of fluid density change with temperature
	or		
	$[M^2/L^3\cdot M_s]$	DRWDU	for solute transport: coefficient of fluid density change with concentration
U_o	[°C]	URHOWO	for energy transport: base temperature for density calculation
	or		
	$[M_s/M]$	URHOWO	for solute transport: base concentration for density calculation

Transport parameters

α_{Lmax_L}	[L]	ALMAX (L)	value of longitudinal dispersivity in direction of k_{max} in each element
α_{Lmin_L}	[L]	ALMIN (L)	value of longitudinal dispersivity in direction of k_{min} in each element
α_{T_L}	[L]	ATAVG (L)	value of transverse dispersivity in each element
σ_w	$[E/(L\cdot °C\cdot s)]$	SIGMAW	for energy transport: fluid thermal conductivity
	or		
	$[m^2/s]$	SIGMAW	for solute transport: molecular diffusivity of solute in fluid

σ_s	$[E/(L\cdot°C\cdot s)]$	SIGMAS	for energy transport: solid grain thermal conductivity (equals zero for solute transport)
c_w	$[E/(M\cdot°C)]$	CW	for energy transport: fluid specific heat capacity (equals one for solute transport)
c_s	$[E/(M\cdot°C)]$	CS	for energy transport: solid grain specific heat capacity (not specified in input data for solute transport)
ρ_s	$[M/L^3]$	RHOS	density of a solid grain in the solid matrix

Reaction and production parameters

Linear Sorption Isotherm

χ_1	$[L_f^3/M_G]$	CHI1	linear distribution coefficient (2.34a) (χ_2 is zero for this isotherm)

Freundlich Sorption Isotherm

χ_1	$[L_f^3/M_G]$	CHI1	Freundlich distribution coefficient (2.35a)
χ_2	[1]	CHI2	Freundlich coefficient (2.35a)

Langmuir Sorption Isotherm

χ_1	$[L_f^3/M_G]$	CHI1	Langmuir distribution coefficient (2.36a)
χ_2	$[L_f^3/M_s]$	CHI2	Langmuir coefficient (2.36a)

Production

γ_1^w	$[s^{-1}]$	PRODF1	for solute transport: rate of first-order production of adsorbate mass in the fluid mass (equals zero for energy transport)
γ_1^s	$[s^{-1}]$	PRODS1	for solute transport: rate of first order production of solute mass in the immobile phase (equals zero for energy transport)

γ_o^w	[(E/M)/s]	PRODF0	for energy transport: zero-order rate of energy production in the fluid
	[$(M_s/M)/s$]	PRODF0	for solute transport: zero-order rate of solute mass production in the fluid
γ_o^s	[$(E/M_G)/s$]	PRODS0	for energy transport: zero-order rate of energy production in the immobile phase
	[$(M_s/M_G)/s$]	PRODS0	for solute transport: zero-order rate of adsorbate mass production in the immobile phase

Boundary conditions and source data

Flow Data - Specified Pressures

NPBC		NPBC	number of nodes at which pressure is a specified constant or function of time
$IPBC_{ipu}$		IPBC(IPU)	node number at which pressure is specified (for all NPBC nodes)
PBC_{ipu}	$[M/(L\cdot s^2)]$	PBC(IPU)	value of specified pressure at node IPBC (for all NPBC nodes)
UBC_{ipu}	[°C]	UBC(IPU)	for energy transport: value of temperature of any fluid which enters the system at node IPBC
	[M_s/M]	UBC(IPU)	for solute transport: value of concentration of any fluid which enters the system at node IPBC

Flow Data - Specified Flows and Fluid Sources

NSOP		NSOP	number of nodes at which a source of fluid mass is specified
$IQCP_{iqp}$		IQCP, IQSOP (IQP)	node number at which a fluid source is specified (for all NSOP nodes)
Q_{IN_i}	[M/s]	QINC, QIN(I)	fluid source rate at source node IQCP (for all nodes)

U_{IN_i}	[°C]	UINC, UIN(I)	for energy transport: value of temperature of any fluid which enters the system at source node IQCP
	or		
	[M_s/M]	UINC, UIN(I)	for solute transport: value of concentration of any fluid which enters the system at source node IQCP

Energy or Solute Data -

Specified Temperatures or Concentrations

NUBC		NUBC	number of nodes at which temperature or concentration is a specified constant or function of time
$IUBC_{ipu}$		IUBC(IPU)	node number at which temperature or concentration is specified (for all NUBC nodes)
UBC	[°C]	UBC(IPU)	for energy transport: value of specified temperature at node IUBC (for all NUBC nodes)
	[M_s/M]	UBC(IPU)	for solute transport: value of specified concentration at node IUBC (for all NUBC nodes)

Energy or Solute Data -

Diffusive Fluxes of Energy or Solute Mass at Boundaries

NSOU		NSOU	number of nodes at which a diffusive energy or solute mass flux (source) is specified
IQCU		IQCU, IQSOU(IQU)	node number at which a flux (source) is specified (for all NSOU nodes)
Ψ_{IN_i}	[E/s]	QUIN(I)	for energy transport: energy flux (source) rate at node IQCU (for all NSOU nodes)
	[M_s/s]	QUIN(I)	for solute transport: solute mass flux (source) rate at node IQCU (for all NSOU nodes)

Initial conditions

t_o	[s]	TSTART	starting time for simulation clock
$p_i(t=t_o)$	[M/(L·s^2)]	PVEC(II)	initial pressure at all nodes in mesh

$U_i(t=t_o)$	[°C]	UVEC(II)	for energy transport: initial temperature at all NN nodes in the mesh
	[M_s/M]	UVEC(II)	for solute transport: initial concentration at all NN nodes in the mesh

Numerical and temporal control data

ν_i	[Ls]	GNU	specified pressure boundary condition 'conductance' factor (4.64)
UP	[1]	UP	fractional upstream weight for asymmetric weighting functions (4.23) (4.24)
Δt	[s]	DELT	initial time step
	[s]	TMAX	maximum allowed simulation time
		ITMAX	maximum allowed number of time steps in a simulation
		ITCYC	time step change cycle
		DTMULT	multiplier for time step change cycle
		DTMAX	maximum time step size allowed when using multiplier
NPCYC		NPCYC	time steps in pressure solution cycle
NUCYC		NUCYC	time steps in temperature or concentration solution cycle
		ITRMAX	maximum number of iterations for nonlinearities per time step
	[M/(L·s^2)]	RPMAX	pressure convergence criterion for iterations
	[°C]	RUMAX	for energy transport: temperature convergence criterion
	[M_s/M]	RUMAX	for solute transport: concentration convergence criterion

Data for options

IREAD	= +1 new simulation - cold start = -1 restart simulation - warm start
ISTORE	= 1 store simulation results for later restart = 0 do not store results

Simulation mode options

SIMULA
- = "SUTRA ENERGY TRANSPORT"
- = "SUTRA SOLUTE TRANSPORT"

IUNSAT
- = 1 allow unsaturated flow
- = 0 saturated flow only

ISSFLO
- = 1 steady-state flow
- = 0 transient flow and transport

ISSTRA
- = 1 steady-state flow and transport
- = 0 transient transport

Velocity Output Option

KVEL
- = 1 output fluid velocity at element centroids
- = 0 no velocity output

Printer Plot Output Option

KPLOTP
- = 1 output of pressure
- = 0 no pressure plots

KPLOTU
- = 1 output plots of temperature or concentration
- = 0 no plots of temperature or concentration

IDIREC
- = +1 plot across page (small)
- = -1 plot along page (large)

NLINPI number of printer lines per inch

NCHAPI	number of printer characters per inch
NCHAPL	number of printer characters per line
PBASE	value for scaling plotted pressures
UBASE	value for scaling plotted temperatures or concentrations

Observation Option

NOBS	number of nodes at which pressure and temperature or concentration will be observed (zero cancels the option)
NTOBS	maximum number of observation time steps
NOBCYC	observations are made every NOBCYC time steps
INOB(I)	observation node numbers

Budget Option

KBUDG	= 1 output fluid mass and energy or solute mass budgets = 0 no budgets

Output Controls

KNODAL	1 output nodewise input data 0 cancel output
KELMNT	1 output elementwise input data 0 cancel output
KINCID	1 output incidence lists 0 cancel output
NPRINT	results are output every NPRINT time steps

7.2 Discretization Rules-of-Thumb

Proper discretization in space and time is the vital factor in obtaining accurate simulation of the physics of flow and transport with a numerical model such as SUTRA. Adequate discretization is vital for two reasons: 1) The ability of a model to represent the variations in system parameters and to simulate complex processes depends on the fineness of discretization. 2) The accuracy and stability of the numerical methods used to represent system processes, in particular, transport, depends on the spatial and temporal discretization. This section describes some general guidelines for designing adequate discretization for simulation with SUTRA.

A 'sufficiently good' discretization allows for accurate simulation of the processes and parameter variations at the scale of interest, and thus the goodness of a discretization is a relative rather than absolute factor. A better discretization is always obtained by making existing discretization finer, but the finer the discretizations are, the more computationally expensive the simulations become.

Relative to a certain adequate level of fineness, even finer discretizations do not practically improve the accuracy of simulation. In contrast, discretization that is too coarse may completely obscure parameter variations and processes of interest in a simulation, and give highly inaccurate results. Unfortunately, simulation results based on inadequate discretization may appear to be a reasonably good representation of flow and transport physics in a particular system. The only way to explicitly check for inadequate discretization of a system is to simulate with a discretization that is assumed to be adequate and then with a significantly finer discretization and compare results. If there are no telling dif-

ferences in the results, then the coarser simulation indeed has been adequately discretized.

Some general guidelines for obtaining adequate discretization, both for parameter representation and for accuracy and stability of numerical methods are given below.

1) Nodes are required where boundary conditions and sources are specified. Should accurate simulation of processes near these specified points be required, then a finer mesh is needed in these areas.

2) A finer mesh is required where parameters vary faster in space. This is often the case near sources or boundary conditions specifying inflows of fluid, solute or energy. The fineness required is that which makes the nodewise, cellwise, or elementwise discretization of the parameter values a good representation of the actual distributions. When a parameter distribution is known a priori, then this discretization is straightforward. However, when the parameter distribution depends on the simulation results then judgement must be exercised in discretization, and the result may be tested by experiment with various discretizations.

It is important to recognize that each node or element does not alone represent a physical entity in an aquifer system. This is demonstrated in the following example which shows that one layer of elements is not a good representation in cross section of a semi-confining layer or aquifer unit. Although permeability is specified elementwise and the permeability of two aquifer units separated by confining layer, viewed in cross-section, is clearly represented visually by three layers of elements, the numerical model does not 'see' three

distinct layers of permeability. Each node at the boundary of these layers experiences some average of the two permeabilities rather than either one. Thus, no node in the system experiences the assigned low permeability of confining layer, and the three-layer discretization is inadequate. More layers of elements are required in each unit to obtain adequate discretization although the model always experiences an average permeability in the elements making up the boundaries of the units. Further refinement of discretization would be required to represent the pressure distribution should accurate simulation of sharply-varying pressures across the confining layer be required.

Discretization of the spatial distribution of transport variables, concentration or temperature, often is that which requires the finest mesh. The spatial distributions of these variables often include a 'front' at which the concentration or temperature changes sharply from high values on one side to low values on the other side. A rule-of-thumb is that at least five elements should divide the front in order to guarantee that the simulated front width arises from simulated physical processes rather than from spreading due to inadequate discretization. When such fronts travel with the flow across a mesh during simulation, then the mesh must be designed fine enough to adequately represent the front at all points along its path. In regions external to the front path, coarser discretization is usually adequate, and an expanding mesh or pinch nodes may be used to design the discretization in this region.

3) The spatial stability of the numerical approximation of the unified transport equation (2.52) depends on the value of a mesh Peclet number, Pe_m, given by:

$$Pe_m = \frac{\varepsilon S_w |\underline{v}| \Delta L_L}{[\varepsilon S_w(\sigma_w + \alpha_L |\underline{v}|) + (1-\varepsilon)\sigma_s]} \quad (7.1)$$

where ΔL_L is the local distance between element sides along a streamline of flow. Spatial instability appears as one or more oscillations in concentration or temperature. Stability is guaranteed in all cases when $Pe_m \leq 2$, which gives a criterion for choosing a maximum allowable element dimension, ΔL_L, along the local flow direction. This criterion significantly affects discretization. Spatial stability is usually obtained with SUTRA when

$$Pe_m \leq 4 \quad (7.2)$$

which gives a less-stringent criterion. Mesh design according to the criterion is critical when concentrations or temperatures change significantly along streamlines, such as when a front is propagated in the direction of flow. When concentrations or temperatures exhibit small changes along streamlines, then the criterion, (7.2) may safely be violated, even by a few orders of magnitude, without inducing spatial instability. An example of this may be cross-sectional simulation of an aquifer containing fresh water and salt water. In such a case, flow often is directed parallel to the front between fresh water and salt water, allowing use of discretization with large mesh Peclet numbers.

In the typical case of solute or energy transport with longitudinal dispersion primarily due to longitudinal mixing, the mesh Peclet number becomes:

$$Pe_m \approx \left(\frac{\Delta L_L}{\alpha_L}\right) \quad (7.3)$$

A discretization rule-of-thumb for simulation with SUTRA which guarantees spatial stability in most cases is:

$$\Delta L_L \leq 4\alpha_L \quad (7.4)$$

While (7.4) deals with adequate discretization for numerical stability it may be interpreted from another point of view. Taken in combination with the considerations of guideline (2) requiring at least five elements across a front, (7.4) implies that a minimum front width which may be simulated when the mesh is designed according to $\Delta L_L \sim 4\alpha_L$ is $20\alpha_L$. Thus for early times following onset of localized energy or solute source, the sharp front that should result may be simulated inaccurately as its width is less than $20\alpha_L$.

4) Discretization for transverse dispersion also may be related to dispersivity. Although an exact guideline is not given, the object of transverse discretization is to make the local element dimension perpendicular to a streamline small relative to the total transverse dispersivity:

$$\Delta L_T < \alpha_T + \frac{1}{|\underline{v}|} \left[\varepsilon S_w \sigma_w + (1-\varepsilon)\sigma_s\right] \qquad (7.5)$$

where ΔL_T is the local element dimension transverse to the flow direction. In the case where the transverse mixing rather than diffusion dominates the transverse dispersion an adequate but stringent rule-of-thumb may be, $\Delta L_T < 10\alpha_T$, although simulation results should be compared for various transverse discretizations.

5) Radial/cylindrical meshes with a well require very fine discretization near the center axis to accommodate the sharply curving pressure distribution. The radial element dimensions may increase outward and become constant at, for example, a size of $4\alpha_L$.

6) Unsaturated flow simulation requires at least as fine discretization as does transport. Spatial instability appears as an oscillation in saturation values. Unsaturated flow parameters may vary sharply in space, especially during wetting events. A rule-of-thumb is to design the mesh to have at least five elements across a saturation front.

7) Discretization in time is done by choosing the size of time steps. Actual time step sizes may be as large as possible while providing adequate discretization of parameter changes in time. As with spatial discretization, the adequacy of a temporal discretization may be tested only by comparing results of simulations carried out with different time step sizes.

For saturated flow simulation, temporal discretization begins with fine time steps which may become significantly larger as the system response slows. The time-step multiplier feature is provided in SUTRA input data to allow this type of temporal discretization.

For unsaturated flow simulation with SUTRA, temporal discretization must be fine enough to keep saturation changes at each node to be small over any time step. A rule-of-thumb is that over a time step, the maximum saturation change is about 0.1.

For transport simulation, temporal changes in concentration or temperature at a point in space are often due to the movement of fronts with the fluid flow. Therefore, adequate discretization of these parameters in time is often related to both fluid velocity and spatial gradients in the parameters. The higher the longitudinal spatial gradient and fluid velocity, the smaller the time step required for adequate temporal discretization. A general guideline is that relatively sharp fronts require time discretization which allows them to move only

a fraction of an element per time step. Broad fronts with low gradient in concentration or temperature have adequate temporal discretization when time steps are chosen to move the front one or more elements per step.

Usually a constant time step size is chosen for transport simulation when flow velocities remain relatively constant during a simulation. For saturated flow and transport, if adequate temporal pressure discretization would allow larger time steps than the temporal transport discretization, then a pressure solution may be done only every n time steps for transport. For example, if the adequate pressure time step is ten times that of transport, then SUTRA input data requires the specification: NPCYC = 10, NUCYC = 1.

7.3 Program Dimensions

All vector and array dimensions in the SUTRA computer code which may vary between simulations are combined for user convenience in three large arrays, RM, RV, and IMV. These arrays are dimensioned by the user in the main routine for SUTRA. No other arrays need be dimensioned. RM contains all of the real matrices in the code, RV contains all of the real vectors, and IMV contains all of the integer matrices and vectors. The dimensions required for these arrays, RMDIM, RVDIM, and IMVDIM, must be specified in the main program to values greater than or equal to those required. The required values are given by relations similar to:

$$RMDIM = 2(NN)(NBI) \qquad (7.6)$$

$$RVDIM = (NNV)(NN) + (NEV+8)(NE) + (NBCN)3 \quad (7.7)$$
$$+ (NOBS+1)(NTOBS+2)2 + NTOBS + 5$$

$$IMVDIM = (NE)8 + NN + (NPINCH)3 + NSOP + NSOU \quad (7.8)$$
$$+ (NBCN)2 + NOBS + NTOBS + 12$$

and

NN = number of nodes

NE = number of elements

NBI = full band width of matrix

NSOP = number of fluid source nodes

NSOU = number of solute or energy source nodes

NPBC = number of specified pressure nodes

NUBC = number of specified U nodes

NBCN = NPBC + NUBC

NPINCH = number of pinch nodes

NOBS = number of observation nodes

NTOBS = number of observation time steps (max)

NNV = number of vectors NN long = approx. 30 (fixed)

NEV = number of vectors NE long = approx. 10 (fixed)

The actual relations and values are listed in the main routine and should be checked there for the most recent SUTRA model version. These dimensions may be greater than but not less than the values given by the relations equivalent to (7.6), (7.7) and (7.8) in the main routine.

7.4 Input and Output Files

The SUTRA computer code requires three or four files to be assigned on the computer in order to run simulations. Two of these are input files and one or two of these are output files.

INPUT FILES:

UNIT-5 A file must be assigned as fortran-unit-5 which contains SUTRA input data for UNIT-5. This file contains all of the data necessary for simulation except initial conditions.

UNIT-55 A file must be assigned as fortran-unit-55 which contains SUTRA input data for UNIT-55. This file contains initial conditions of pressure and concentration or temperature for the simulation to be done.

OUTPUT FILES:

UNIT-6 A file must be assigned as fortran-unit-6 on which printed output of the simulation will be placed.

UNIT-66 An optional output file must be assigned as fortran-unit-66 if the option to save the solution of the most recently completed time step for later restart is chosen in UNIT-5 when (ISTORE = 1). Data will be written to this file in a format equivalent to UNIT-55 data so that this file may later be used as UNIT-55.

The data lists and formats for the input files are given in section 7.7, "SUTRA Input Data List."

7.5 User-Supplied Programming

When SUTRA is used for simulation of systems with unsaturated flow, then the user must code the desired unsaturated flow functions in subroutine UNSAT. When SUTRA simulation includes time-dependent boundary conditions or sources, then the desired temporal variations must be coded by the user in subroutine BCTIME.

Subroutine UNSAT

The general operation of this subroutine is described in section 5.7, "Program Structure." Given a single value of pressure, UNSAT must provide values of S_w, $(\partial S_w/\partial p)$, and k_r. UNSAT consists of three sections. The user must supply code in each of these sections. An example using the unsaturated flow functions (2.8), (2.11), and (2.21a) and (2.21b) is given in the listing of Subroutine UNSAT in APPENDIX A, "SUTRA Program Listing."

The first section requires specification of saturation, S_w, as a function of pressure, p. The second section requires specification of the derivative of saturation with respect to pressure, p, or saturation, S_w. The third section requires specification of the relative permeability, k_r, as a function of either saturation, S_w, or pressure, p. The pressure value which is passed to UNSAT is the projected value, the most recent iterate or the newly obtained solution. The values are either at Gauss points or at nodes.

Any convenient programming algorithm may be used to implement these functions in UNSAT. Some possibilities are: use of explicit expressions, as in the example; use of data statements; use of logical statements to give piecewise continuous

functions; or use of READ statements to input new data to the functions from either UNIT-5 or a new data file. Sometimes functions with entry pressure or residual saturation require that functions used be switched when passing by these values. Logical statements which check S_w or p values may be used to switch to other functions or to constant values, as required.

Subroutine BCTIME

The general operation of this subroutine is described in section 5.7, "Program Structure." At the beginning of each time step, BCTIME must provide: values of all specified time-varying pressure values and temperature or concentration values of fluid inflow at these nodes; values of specified time-varying temperature or concentration; values of specified time-varying fluid sources (or sinks) and temperatures or concentrations of these flows if they are inflows; and values of time of time-varying energy or solute mass sources (or sinks). BCTIME consists of four sections, each dealing with one of the above types of specification. The user must supply code in the section (or sections) of BCTIME which specifies the particular type of time-varying boundary condition or source desired.

The first section is used for specifying either time variation of pressure, or time variation of the temperature or concentration of any fluid which enters the system at a point of specified pressure, or both. The coding must be entered within a loop which checks all NPBC specified pressure nodes for the time-variability flag. This flag is a negative node number in the list of specified pressure nodes IPBC(IP). The counter for the list is IP. When the loop finds that the IP^{th} node number, IPBC(IP), is negative, then the actual node number is given by I = -IPBC(IP). In this case, the user must supply code which specifies a value

appropriate for the current time step, for both PBC(IP), which is the specified pressure for the IP^{th} specified pressure node (node I), and for UBC(IP), which is the specified temperature or concentration of any inflow at the IP^{th} specified pressure node (node I). The loop skips over the positive node numbers in the list IPBC(IP).

The second section is used for specifying time variation of temperature or concentration. The coding must be entered within a loop which checks all NUBC specified temperature or concentration (U) nodes for the time-variability flag. This flag is a negative node number in the list of specified U nodes, IUBC(IU). The list begins in the $(NPBC + 1)^{th}$ element of IUBC as shown in the description of subroutine BOUND in section 5.7, "Program Structure." The first NPBC elements of IUBC are blank. The counter for the list is IU. If the loop finds that the IU^{th} node number, IUBC(NPBC + IU), is negative, then the actual node number is given by I = -IUBC(NPBC + IU). In this case, the user must supply code which specifies a value, appropriate for the current time step, for UBC(NPBC + IU), which is the specified temperature or concentration for the IU^{th} specified U node (node I). The loop skips over node numbers, IUBC(NPBC + IU), which are positive.

The third section is used for specifying time variation of either fluid sources (or sinks), temperature or concentration of inflowing fluid at sources, or both. The coding must be entered within a loop which checks all NSOP fluid source nodes for the time-variability flag. This flag is a negative node number in the list of fluid source nodes, IQSOP(IQP). The counter for the list is IQP. If the loop finds that the IQP^{th} node number IQSOP(IQP), is negative, then the actual node number is given by I = -IQSOP(IQP). In this case, the user must supply code which specifies a value appropriate for the current time step, for

both QIN(I), which is the specified fluid source for node I (the IQPth specified fluid source node), and for UIN(I), which is the temperature or concentration of inflowing fluid at node I. The loop skips over node numbers in the list, IQSOP(IQP), which are positive.

The fourth section is used for specifying time variation of energy or solute mass sources. The coding must be entered within a loop which checks all NSOU specified energy or solute mass source nodes for the time-variability flag. This flag is a negative node number in the list of specified energy or solute mass source nodes, IQSOU(IQU). The counter for the list is IQU. If the loop finds that the IQUth node number, IQSOU(IQU), is negative, then the actual node number is given by I = -IQSOU(IQU). In this case, the user must supply code which specifies a value appropriate for the current time step, for QUIN(I), which is the specified energy or solute mass source for node I (the IQUth specified energy or solute mass source node). The loop skips over the positive node numbers in the list, IQSOU(IQU).

The current time at the end of the present time step in seconds, TSEC, and in other time units is available for use in specifying time variations. Any convenient programming algorithm may be used to implement the time-variations in BCTIME. Some possibilities are: use of expressions as explicit functions of time such as, for example, a sine function to represent tidal pressure variations; use of data statements and new arrays explicitly dimensioned in BCTIME; use of logical statements to give stepped or piecewise continuous functions; or use of READ statements to input the time-varying values directly from SUTRA UNIT-5 or a new data file. If different functions or values are to be specified at various nodes, then the user must also supply code to distinguish which functions apply to which specified node numbers.

7.6 Modes and Options

Simulation modes

SUTRA may simulate flow and transport in three temporal modes for either energy or solute transport. The modes are: (1) transient flow and transport, (2) steady flow with transient transport, and (3) steady flow and steady transport, where mode (1) is the most computationally expensive and mode (3), the least expensive. Modes (2) and (3) are not applicable to all problems. The classes of problems amenable to solution by each node is given below.

(1) Transient Flow and Transient Transport

Allows for simulation of any physical problem which SUTRA deals with: either saturated or unsaturated flow or both; variable fluid density and viscosity; any sorption isotherm; energy or solute transport.

(2) Steady-State Flow and Transient Transport

Allows for simulation of a restricted class of SUTRA problems: saturated flow only; constant fluid density and viscosity; any sorption isotherm; energy transport with only small variations in temperature, or solute transport.

(3) Steady-State Flow and Steady-State Transport

Allows for simulation of the most restricted class of SUTRA problems: saturated flow only; constant fluid density and viscosity; linear sorption isotherm only; energy transport with only small variations in temperature, or solute transport.

These modes are specified in UNIT-5 input data by the values of ISSFLO, ISSTRA, and SIMULA.

Output options

A number of output options are available which help to interpret SUTRA simulation results. These are: (1) printer plots, (2) velocity output, (3) budget output, and (4) observation node output. The first three options require some extra computations and should be used only when necessary, as the extra calculations are done for each printed output.

(1) Printer Plots

Plots are available which are printed on each time step on which there is output. The plot is a map of three-digit pressures, temperatures or concentrations at the nodes which may be contoured by hand for an initial view of simulation results. Either a pressure plot or temperature (concentration) plot is output, or both on each time step with output. The plot consists of three significant figures of the pressure or temperature (or concentration) value at each node printed approximately at the nodal location in a map scaled to the printer paper. The map may be oriented either across the output page for a small plot, or along the page for a large plot. A plot of the locations of node numbers is provided with the input data print-out. Unfortunately, when some nodes in the mesh are grouped closely relative to the others, the printed three digits at clustered nodes may overlap and obscure the values. This typically occurs near the center axis for

meshes in cylindrical coordinates. Use of the large plot may separate the values but the plot size can become unwieldy. Computer graphics contouring must then be employed, and is clearly more convenient than hand-contoured printer plots when available.

(2) Velocity Output

An output of fluid velocity is available, the information in which may be used to plot velocity vectors everywhere in the simulated spatial region with computer graphics software supplied by the user. These velocities are calculated and output on each time step that a pressure solution is output. One velocity is calculated in each finite element, at the location of the element centroid, as described in section 5.5, "Velocity Calculation for Output." Velocity output occurs in two groups of values: first, the magnitude of the velocity vector at each element centroid, and second, the angle measured (with a counter-clockwise positive value) from the positive x-axis to the velocity vector direction. Note that velocity values are lagged one time step if a non-iterative solution is used. (In this case, they are calculated not with the new pressure solution, but with the solution of the previous time step and with fluid density values of the step before that. This keeps the velocity calculations consistent in time.) This option is controlled by UNIT-5 parameter, KVEL.

(3) Budget Output

A fluid mass and energy or solute mass budget output is available as an aid in tracking the simulated behavior of a system. The budget is not a check on numerical accuracy of the model as the calculations involved in determining the budget are less accurate than the calcula-

tions used to carry out the SUTRA simulation. The budget is output on each time step with printer output, and tallies total system changes in fluid mass, and energy or solute mass for the time step. Besides the totals of these quantities for the entire simulated region, the budget lists time step total gains or losses in these quantities at each specified pressure node, fluid source node, and energy or solute mass source node in the mesh. More information about the budget calculations is given in section 5.6, "Budget Calculations." The option is controlled by UNIT-5 parameter, KBUDG.

(4) Observation Node Output

An observation node output is available which observes pressure and temperature or concentration at particular nodes in the system during the simulation, and outputs the observations in table form after the last time step of the simulation has been completed. For each observed node, the table consists of three columns of numbers: the time of the observation, the observed pressure value, and the observed temperature or concentration value. Any number of observation nodes (NOBS) may be chosen, and observations may be requested every NOBCYC time steps.

(p. 247 follows)

7.7 SUTRA Input Data List

List of Input Data for UNIT 5

Model Series: SUTRA
Model Version: V1284-2D

Note that three arrays in the main routine of the code need to be dimensioned. The procedure for choosing dimensions is listed in the main routine itself, near the place where the dimensions need be specified.

DATASET 1: Input Data Heading (one card)

Variable	Format	Description
SIMULA	2A6	For energy transport simulation, write "SUTRA ENERGY TRANSPORT". For solute transport simulation, write "SUTRA SOLUTE TRANSPORT". The rest of the card is not used by SUTRA and may either be left blank or may be used to note an additional label for this UNIT 5 data list.

DATASET 2: Output Heading (two cards)

Variable	Format	Description
TITLE1	80A1	First line of a heading for the input data set.
TITLE2	80A1	Second line of heading for the input data set.
		These two lines are printed as a heading on SUTRA output.

DATASET 3: Simulation Control Numbers (one card)

Variable	Format	Description
NN	I5	Exact number of nodes in finite element mesh.
NE	I5	Exact number of elements in finite element mesh.
NBI	I5	Full bandwidth of global banded matrix. NBI is equal to one plus twice maximum difference in node numbers in the element containing the largest node number difference in the mesh. This number is critical to computational efficiency, and should be minimized by careful numbering of the nodes (see Figure 7.1). Setting NBI too small causes SUTRA to automatically print out the correct value and stop.
NPINCH	I5	Exact number of pinch nodes in the finite element mesh.
NPBC	I5	Exact number of nodes at which pressure is a specified constant value or function of time.
NUBC	I5	Exact number of nodes at which temperature or concentration is a specified constant value or function of time.
NSOP	I5	Exact number of nodes at which a fluid source/sink is a specified constant value or function of time.
NSOU	I5	Exact number of nodes at which an energy or solute mass source/sink is a specified constant value or function of time.
NOBS	I5	Exact number of nodes at which observations will be made. Set to zero for no observations.
NTOBS	I5	Maximum number of time steps on which observations will be made. This depends on both the number of time steps in the simulation (DATASET 6), and on the frequency of observations (DATASET 21). NTOBS may be set to a value greater than that needed. Set to zero for no observations.

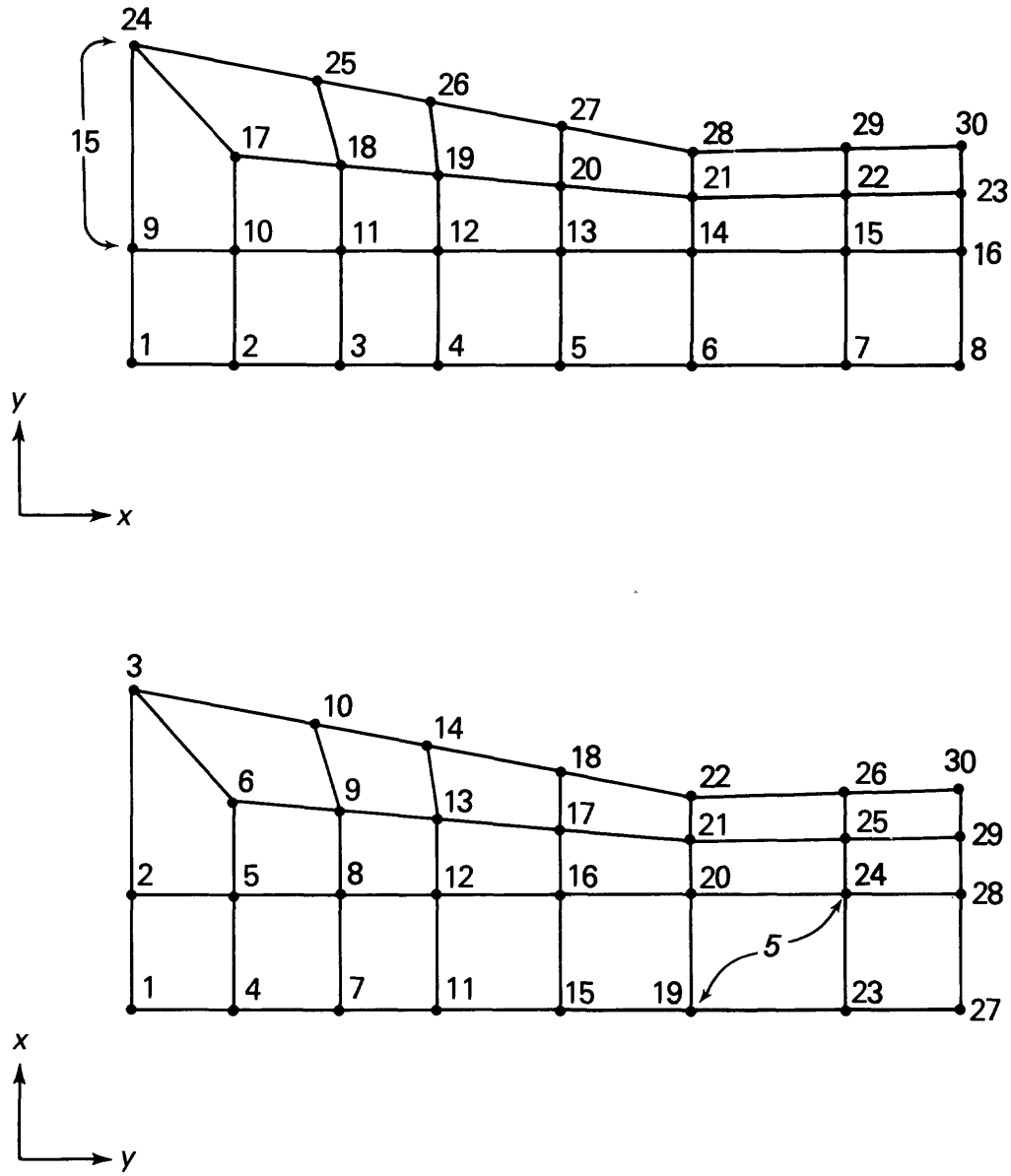


Figure 7.1
Minimization of band width by careful numbering of nodes.

DATASET 4: Simulation Mode Options (one card)

Variable	Format	Description
IUNSAT	I5	Set to +1 to allow simulation of unsaturated and saturated flow. Set to 0 to allow simulation of only saturated flow. When unsaturated flow is allowed (IUNSAT = 1) then the unsaturated flow functions must be programmed by the user in Subroutine UNSAT.
ISSFLO	I5	Set to 0 for simulation with TRANSIENT groundwater flow. Set to +1 for simulation with STEADY-STATE groundwater flow. If fluid density is to change with time, then TRANSIENT flow must be selected.
ISSTRA	I5	Set to 0 for simulation with TRANSIENT solute or energy transport. Set to +1 for simulation of STEADY-STATE transport. Note that steady-state transport requires a steady-state flow field. So, if ISSTRA = +1, then, also set ISSFLO = +1
IREAD	I5	To read initial condition data (UNIT 55) for cold start (first time step of a simulation), set to +1. To read initial condition data (UNIT 55) for simulation restart (to read data which has previously been stored by SUTRA on UNIT 66), set to -1.
ISTORE	I5	To store results of most recently completed time step on UNIT 66 for later use as initial conditions on a restart, set to + 1. To cancel storage, set to 0. This option is recommended as a backup for storage of results of intermediate time steps during long simulations. Should the execution halt unexpectedly, it may be restarted with initial conditions consisting of results of the last successfully completed time step stored on UNIT 66.

DATASET 5: Numerical Control Parameters (one card)

Variable	Format	Description
UP	G10.0	Fractional upstream weight for stabilization of oscillations in results due to highly advective transport or unsaturated flow. UP may be given any value from 0.0 to +1.0. UP = 0.0 implies no upstream weighting (Galerkin method). UP = 0.5 implies 50% upstream weighting. UP = 1.0 implies full (100%) upstream weighting. Recommended value is zero. Warning: upstream weighting increases the local effective longitudinal dispersivity of the simulation by approximately (UP·(ΔL)/2) where ΔL is the local distance between element sides along the direction of flow. Note that the amount of this increase varies from place to place depending on flow direction and element size. Thus a non-zero value for UP actually changes the value of longitudinal dispersivity used by the simulation, and also broadens otherwise sharp saturation fronts.
GNU	G15.0	Pressure boundary condition, 'conductance'. A high value causes SUTRA simulated pressure and specified pressure values at specified pressure nodes to be equal in all significant figures. A low value causes simulated pressures to deviate significantly from specified values. The ideal value of GNU causes simulated and specified pressures to match in the largest six or seven significant figures only, and deviate in the rest. Trial-and-error is required to determine an ideal GNU value for a given simulation by comparing specified pressures with those calculated at the appropriate nodes for different values of GNU. An initial guess of 0.01 is suggested.

DATASET 6: Temporal Control and Solution Cycling Data (one card)

Variable	Format	Description	
ITMAX	I5	Maximum allowed number of time steps in simulation.	
DELT	G15.0	Duration of initial time step. [s]	
TMAX	G15.0	Maximum allowed simulation time. [s] SUTRA time units are always in seconds. Other time measures are related as follows: [min] = 60. [s] [h] = 60. [min] [d] = 24. [h] [week] = 7. [d] [mo] = 30.4375 [d] [yr] = 365.250 [d]	
ITCYC	I10	Number of time steps in time step change cycle. A new time step size is begun at time steps numbered: 1+ n (ITCYC).	
DTMULT	G10.0	Multiplier for time step change cycle. New time step size is: (DELT)(DTMULT).	
DTMAX	G15.0	Maximum allowed size of time step when using time step multiplier. Time step size is not allowed to increase above this value.	
NPCYC	I5	Number of time steps in pressure solution cycle. Pressure is solved on time steps numbered: n(NPCYC), as well as on initial time step.	Either NPCYC or NUCYC must be set to 1.
NUCYC	I5	Number of time step in temperature/ concentration solution cycle. Transport equation is solved on time steps numbered: n(NUCYC) as well as on initial time step.	

DATASET 7: Output Controls and Options (one card)

Variable	Format	Description
NPRINT	I5	Printed output is produced on time steps numbered: n(NPRINT), as well as on first and last time step.
KNODAL	I5	A value of 0 cancels printout of node coordinates, nodewise element thicknesses, and nodewise porosities. Set to +1 for full printout.
KELMNT	I5	A value of 0 cancels printout of element-wise permeabilities and elementwise dis-persivities. Set to +1 for full printout.
KINCID	I5	A value of 0 cancels printout of node incidences and pinch node incidences in elements. Set to +1 for full printout.
KPLOTP	I5	Set to a value of +1 for contourable printer plot of pressures at all nodes in mesh. Set to 0 to cancel pressure plot.
KPLOTU	I5	Set to a value of +1 for contourable printer plot of concentrations or temperatures at all nodes in mesh. Set to 0 to cancel plot.
KVEL	I5	Set to a value of +1 to calculate and print fluid velocities at element centroids each time printed output is produced. Note that for non-steady state flow, velocities are based on results and pressures of the previous time step or iteration and not on the newest values. Set to 0 to cancel option.
KBUDG	I5	Set to a value of +1 to calculate and print a fluid mass budget and energy or solute mass budget each time printed output is produced. A value of 0 cancels the option.

DATASET 8: Iteration Controls (one card)

Variable	Format	Description
ITRMAX	I10	Maximum number of iterations allowed per time step to resolve non-linearities. Set to a value of +1 for non-iterative solution. Non-iterative solution may be used for saturated aquifers when density variability of the fluid is small, or for unsaturated aquifers when time steps are chosen to be small
RPMAX	G10.0	Absolute iteration convergence criterion for pressure solution. Pressure solution has converged when largest pressure change from the previous iteration's solution of any node in mesh is less then RPMAX. May be left blank for non-iterative solution.
RUMAX	G10.0	Absolute iteration convergence criterion for transport solution. Transport solution has converged when largest concentration on temperature change from the previous iteration's solution of any node in mesh is less than RUMAX. May be left blank for non-iterative solution.

DATASET 9: Fluid Properties (one card)

Variable	Format	Description
COMPFL	G10.0	Fluid compressibility, $\beta=(1/\rho)(\partial\rho/\partial p)$. $\lvert M/(L\cdot s^2)\rvert^{-1}$. Note, specific pressure storativity is: $S_{op} = (1-\varepsilon)\alpha + \varepsilon\beta$
CW	G10.0	Fluid specific heat, c_w. $\lvert E/(M\cdot{}^\circ C)\rvert$ (May be left blank for solute transport simulation.)
SIGMAW	G10.0	Fluid diffusivity, σ_w. For energy transport represents fluid thermal conductivity, $\lvert E/(L\cdot{}^\circ C\cdot s)\rvert$. For solute transport represents molecular diffusivity of solute in pure fluid. $\lvert L^2/s\rvert$.
RHOWØ	G10.0	Density of fluid at base concentration or temperature. $\lvert M/L^3\rvert$.
URHOWØ	G10.0	Base value of solute concentration (as mass fraction) or temperature of fluid at which base fluid density, RHOWØ is specified. $\lvert M_s/M\rvert$ or $\lvert{}^\circ C\rvert$.
DRWDU	G10.0	Fluid coefficient of density change with concentration (fraction) or temperature: ρ = RHOWØ + DRWDU (U-URHOWØ). $\lvert M/(L^3\cdot M_s)\rvert$ or $\lvert M/(L^3\cdot{}^\circ C)\rvert$
VISCØ	G10.0	For solute transport: fluid viscosity, μ, $\lvert M/L\cdot s\rvert$. For energy transport. this value is a scale factor. It multiplies the viscosity which is calculated internally in units of $\lvert kg/m\cdot s\rvert$. VISCØ may be used for energy transport to convert units of $\lvert kg/m\cdot s\rvert$ to desired units of viscosity.

DATASET 10: Solid Matrix Properties (one card)

Variable	Format	Description
COMPMA	G10.0	Solid matrix compressibility, $\alpha=(1-\varepsilon)^{-1}\ \partial\varepsilon/\partial p$. $[M/(L\cdot s^2)]^{-1}$
CS	G10.0	Solid grain specific heat, c_s. $[E/(M\cdot°C)]$ (May be left blank for solute transport simulation.)
SIGMAS	G10.0	Solid grain diffusivity, σ_s. For energy transport represents thermal conductivity of a solid grain. $[E/(L\cdot°C\cdot s)]$ (May be left blank for solute transport simulation.)
RHOS	G10.0	Density of a solid grain, ρ_s. $[M/L^3]$

DATASET 11: Adsorption Parameters (one card)

Variable	Format	Description
ADSMOD	A10	For no sorption or for energy transport simulation write "NONE" beginning in column one, and leave rest of card blank. For linear sorption model, write "LINEAR" beginning in column one. For Freundlich sorption model write "FREUNDLICH" beginning in column one. For Langmuir sorption model write "LANGMUIR" beginning in column one.
CHI1	G10.0	Value of linear, Freundlich or Langmuir distribution coefficient, depending on sorption model chosen as ADSMOD, χ_1. $\lvert L_f^3/M_G \rvert$.
CHI2	G10.0	Value of Freundlich or Langmuir coefficient, depending on sorption model chosen as ADSMOD. Leave blank for linear sorption. χ_2. $\lvert 1 \rvert$ for Freundlich. $\lvert L_f^3/M_s \rvert$ for Langmuir.

DATASET 12: Production of Energy or Solute Mass (one card)

Variable	Format	Description
PRODF0	G10.0	Zero-order rate of production in the fluid γ_0^w. $[(E/M)/s]$ for energy production, $[(M_s/M)/s]$ for solute mass production.
PRODS0	G10.0	Zero-order rate of production in the immobile phase, γ_0^s. $[(E/M_G)/s]$ for energy production, $[(M_s/M_G)/s]$ for adsorbate mass production.
PRODF1	G10.0	First-order rate of solute mass production in the fluid, γ_1^w. $[s^{-1}]$ Leave blank for energy transport.
PRODS1	G10.0	First-order rate of adsorbate mass production in the immobile phase, γ_1^s. $[s^{-1}]$ Leave blank for energy transport.

DATASET 13: Orientation of Coordinates to Gravity (one card)

Variable	Format	Description
GRAVX	G10.0	Component of gravity vector in +x direction. $[L^2/s]$ GRAVX = $-\|\underline{g}\|$ (∂ELEVATION/∂x), where $\|\underline{g}\|$ is the total acceleration due to gravity in $[L^2/s]$.
GRAVY	G10.0	Component of gravity vector in +y direction. $[L^2/s]$ GRAVY = $-\|\underline{g}\|$ (∂ELEVATION/∂y), where $\|\underline{g}\|$ is the total acceleration due to gravity in $[L^2/s]$.

DATASET 14A: Scale Factor for Nodewise Data (one card)

Variable	Format	Description
	5X	In the first five columns of this card write "NODE ", leaving one column blank.
SCALX	G10.0	The scaled x-coordinates of nodes in DATASET 14B are multiplied by SCALX in SUTRA. May be used to change from map to field scales, or from English to SI units. A value of 1.0 gives no scaling.
SCALY	G10.0	The scaled y-coordinates of nodes in DATASET 14B are multiplied by SCALY in SUTRA. May be used to change from map to field scales, or from English to SI units. A value of 1.0 gives no scaling.
SCALTH	G10.0	The scaled element (mesh) thicknesses at nodes in DATASET 14B are multiplied by SCALTH in SUTRA. May be used to easily change entire mesh thickness or to convert English to SI units. A value of 1.0 gives no scaling.
PORFAC	G10.0	The scaled nodewise porosities of DATASET 14B are multiplied by PORFAC in SUTRA. May be used to easily assign a constant porosity value to all nodes by setting PORFAC=porosity, and all POR(II)=1.0 in DATASET 14B.

DATASET 14B: Nodewise Data (one card for each of NN nodes)

Variable	Format	Description
II	I5	Number of node to which data on this card refers, i.
X(II)	G10.0	Scaled x-coordinate of node II, x_i. [L]
Y(II)	G10.0	Scaled y-coordinate of node II, y_i. [L]
THICK(II)	G10.0	Scaled thickness of mesh at node II. [L] In order to simulate radial cross-sections, set THICK(II) = $(2\pi)(\text{radius}_i)$, where radius_i is the radial distance from the vertical center axis to node i.
POR(II)	G10.0	Scaled porosity value at node II, ε_i. [1]

DATASET 15A: Scale Factors for Elementwise Data (one card)

Variable	Format	Description
	10X	In the first ten columns of this card write "ELEMENT ", leaving three columns blank.
PMAXFA	G10.0	The scaled maximum permeability values of elements in DATASET 15B are multiplied by PMAXFA in SUTRA. May be used to convert units or to aid in assignment of maximum permeability values in elements.
PMINFA	G10.0	The scaled minimum permeability values of elements in DATASET 15B are multiplied by PMINFA in SUTRA. May be used to convert units or to aid assignment of minimum permeability values in elements.
ANGFAC	G10.0	The scaled angles between the maximum permeability direction and the x-axis of elements in DATASET 15B are multiplied by ANGFAC in SUTRA. May be used to easily assign a uniform direction of anisotropy by setting ANGFAC= angle, and all ANGLEX(L)=1.0 in DATASET 15B.
ALMAXF	G10.0	The scaled maximum longitudinal dispersivities of elements in DATASET 15B are multiplied by ALMAXF in SUTRA. May be used to convert units or to aid in assignment of dispersivities.
ALMINF	G10.0	The scaled minimum longitudinal dispersivities of elements in DATASET 15B are multiplied by ALMINF in SUTRA. May be used to convert units or to aid in assignment of dispersivities.
ATAVGF	G10.0	The scaled average transverse dispersivities of elements in DATASET 15B are multiplied by ATAVGF in SUTRA. May be used to convert units or to aid in assignment of dispersivity.

DATASET 15B: Elementwise Data (one card for each of NE elements)

Variable	Format	Description
L	I10	Number of element to which data on this card refers.
PMAX(L)	G10.0	Scaled maximum permeability value of element L, $k_{max}(L)$. $[L^2]$
PMIN(L)	G10.0	Scaled minimum permeability value of element L, $k_{min}(L)$. $[L^2]$ Isotropic permeability requires: PMIN(L)=PMAX(L).
ANGLEX(L)	G10.0	Angle measured in counterclockwise direction from +x-direction to maximum permeability direction in element L, θ_L. [°] Arbitrary when both PMIN(L)=PMAX(L), and ALMAX(L) = ALMIN(L).
ALMAX(L)	G10.0	Scaled longitudinal dispersivity value of element L in the direction of maximum permeability PMAX(L), $\alpha_{Lmax}(L)$. [L]
ALMIN(L)	G10.0	Scaled longitudinal dispersivity value of element L in the direction of minimum permeability PMIN(L), $\alpha_{Lmin}(L)$. [L]
ATAVG(L)	G10.0	Scaled average transverse dispersivity value of element L, $\alpha_T(L)$. [L]

DATASET 16: Data for Printer Plot (Two or three cards when plot has been requested by DATASET 7)

O M I T when no plot is requested

Variable	Format	Description
Card 1: (always required when plot is requested)		
IDIREC	I5	Chooses plot direction: Set to -1 for small plot which fits across the output page. Set to +1 for larger plot which is oriented along the output page.
NLINPI	I5	Number of printer lines per inch.
NCHAPI	I5	Number of printer characters per inch.
NCHAPL	I5	Number of printer characters per output line. The plotting routine prints three digits of the nodal value to be plotted at the (x,y) location of the node on a map of the mesh which the routine constructs. The three digits are not necessarily the first three digits of the value to be plotted, but are always one digit to the left and two digits to the right of the decimal point. Thus, if the value to be plotted is 1234.567, then the digits 456, are printed at the nodal location on the output.
Card 2: (include this card only when pressure plots are requested in DATASET 7)		
PBASE	G13.0	Value for scaling plotted pressures. The pressure value to be plotted, p_{PLOT}, is calculated by SUTRA as p_{PLOT} = (true pressure p_i/PBASE) PBASE should be used to scale out powers of ten and to shift the scaled digits of interest to the position of the three plotted digits.

Variable	Format	Description
Card 3: (include this card only when temperature or concentration plots are requested in DATASET 7)		
UBASE	G13.0	Value for scaling plotted temperature or concentration values. The value to be plotted U_{PLOT}, is calculated by SUTRA as: U_{PLOT}= (true value U_i/UBASE). For example, UBASE may be set to one-tenth of the highest source concentration in the system; then fractional concentrations relative to the highest concentration are plotted with digits ranging from 000 to 999 which represents a relative concentration of 1.000 (~0.999).

DATASET 17: Data for Fluid Source and Sinks (one card for each of NSOP fluid source nodes as specified in DATASET 3, plus one blank card)

O M I T when there are no fluid source nodes

Variable	Format	Description
IQCP	I10	Number of node to which source/sink data on this card refers. Specifying the node number with a negative sign indicates to SUTRA that the source flow rate or concentration or temperature of the source fluid vary in a specified manner with time. Information regarding a time-dependent source node must be programmed by the user in Subroutine BCTIME, and should not be included on this card.
QINC	G15.0	Fluid source (or sink) which is a specified constant value at node IQCP, Q_{IN}. |M/s| A positive value is a source of fluid to the aquifer. Leave blank if this value is specified as time-dependent in Subroutine BCTIME. Sources are allocated by cell as shown in Figure 7.2 for equal-sized elements. For unequal-sized elements, sources are allocated in proportion to the cell length, area or volume over which the source fluid enters the system.
UINC	G15.0	Temperature or solute concentration (mass fraction) of fluid entering the aquifer which is a specified constant value for a fluid source at node IQCP, U_{IN}. |°C| or |M_s/M| Leave blank if this value is specified as time-dependent in Subroutine BCTIME.

Last card:

B L A N K C A R D	Placed immediately following all NSOP fluid source node cards.

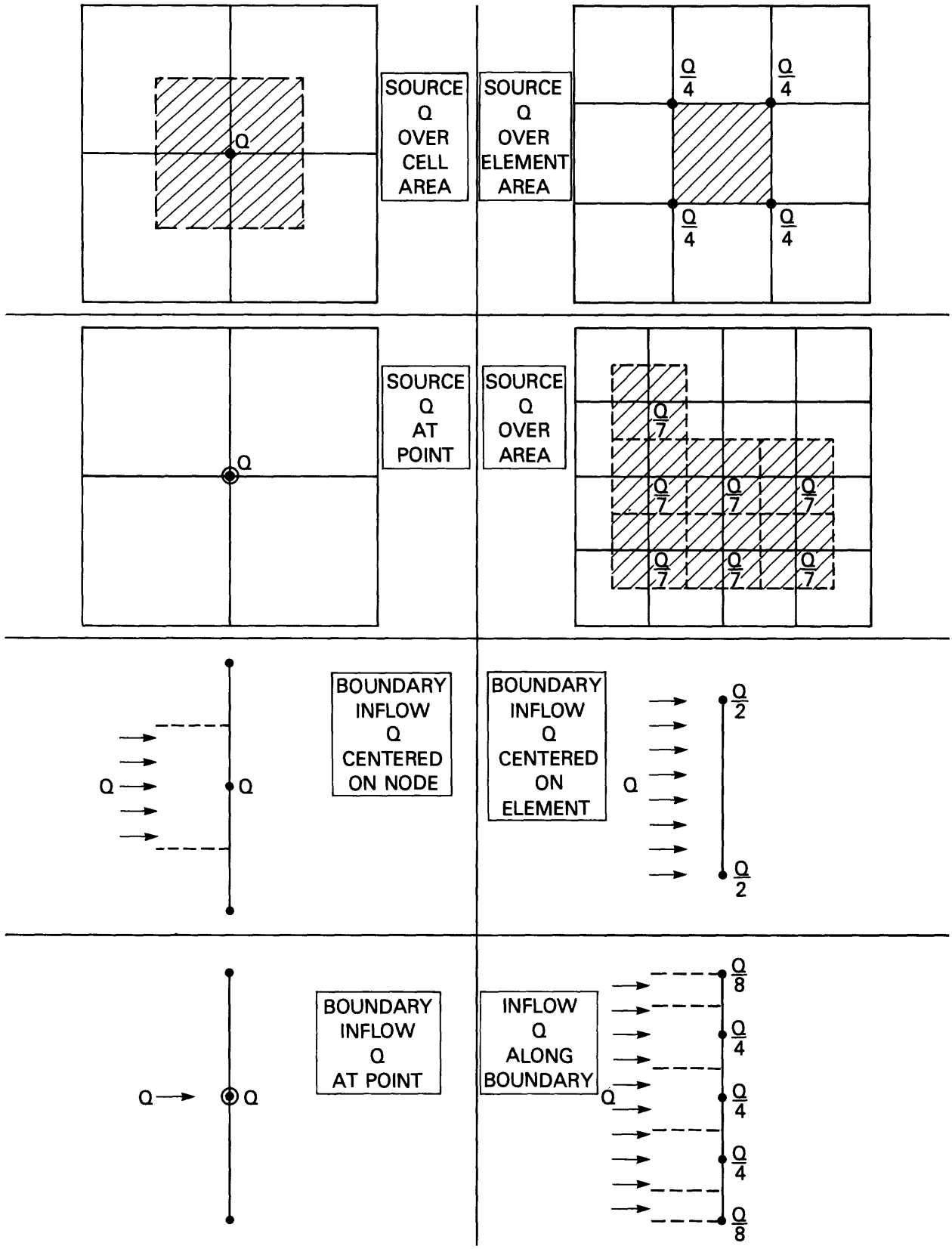


Figure 7.2
Allocation of sources and boundary fluxes in equal-sized elements.

DATASET 18: Data for Energy or Solute Mass Sources and Sinks
(one card for each NSOU energy or solute source nodes as specified in DATASET 3, plus one blank card)

O M I T when there are no energy or solute source nodes

Variable	Format	Description
IQCU	I10	Number of node to which source/sink data on this card refers. Specifying the node number with a negative sign indicates to SUTRA that the source rate varies in a specified manner with time. All information regarding a time-dependent source node must be programmed by the user in Subroutine BCTIME, and a value should not be included in this card. Sources are allocated by cell as shown in Figure 7.2 for equal-sized elements. For unequal-sized elements, sources are allocated in proportion to the cell length, area or volume over which the source energy or solute mass enters the system.
QUINC	G15.0	Source (or sink) which is a specified constant value at node IQCU, Ψ_{IN}. |E/s| for energy transport, $\|M_s/s\|$ for solute transport. A positive value is a source to the aquifer. Leave blank if IQCU is negative, and this value is specified as time-dependent in Subroutine BCTIME.

Last card:

B L A N K C A R D	Placed immediately following all NSOU energy or solute mass source node cards.

DATASET 19: Data for Specified Pressure Nodes (one card for each of NPBC specified pressure nodes as indicated in DATASET 3, plus one blank card)

O M I T when there are no specified pressure nodes

Variable	Format	Description
Cards 1 to NPBC:		
IPBC	I5	Number of node to which specified pressure data on this card refers. Specifying the node number with a negative sign indicates to SUTRA that the specified pressure value or inflow concentration or temperature at this node vary in a specified manner with time. Information regarding a time-dependent specified pressure node must be programmed by the user in Subroutine BCTIME, and should not be included on this card.
PBC	G20.0	Pressure value which is a specified constant at node IPBC. $\lvert M/(L \cdot s^2) \rvert$ Leave blank if this value is specified as time-dependent in Subroutine BCTIME.
UBC	G20.0	Temperature or solute concentration of any external fluid which enters the aquifer at node IPBC. UBC is a specified constant value. $\lvert {}^{\circ}C \rvert$ or $\lvert M_s/M \rvert$ Leave blank if this value is specified as time-dependent in Subroutine BCTIME.
Last card:		
B L A N K C A R D		Placed immediately following all NPBC specified pressure cards.

DATASET 20: Data for Specified Concentration or Temperature Nodes
(one card for each of NUBC specified concentration or temperature nodes indicated in DATASET 3, plus one blank card)

O M I T when there are no specified concentration or temperature nodes

Variable	Format	Description
Cards 1 to NUBC:		
IUBC	I5	Number of node to which specified concentration or temperature data on this card refers. Specifying the node number with a negative sign indicates to SUTRA that the specified value at this node varies in a specified manner with time. This time-dependence must be programmed by the user in Subroutine BCTIME, and a value should not be included on this card.
UBC	G20.0	Temperature or solute concentration value which is a specified constant at node IUBC. [°C] or [M_s/M] Leave blank if IUBC is negative and this value is specified as time-dependent in Subroutine BCTIME.
Last card:		
B L A N K C A R D		Placed immediately following all NUBC specified temperature or concentration cards.

DATASET 21: Observation Node Data (one card plus one card for each (NOBS+16)/16 (integer arithmetic) observation nodes as specified in DATASET 3)

O M I T when there are no observation nodes

Variable	Format	Description
Card 1:		
NOBCYC	I10	Observations of pressure and temperature or concentration will be made at all observation nodes specified below every NOBCYC time steps.
Cards 2 to (NOBS+16)/16		
INOB	16I5	Node numbers of observation nodes. (Sixteen nodes per card.) Enter a value of zero as an extra observation node number following the last real observation node in order to indicate to SUTRA that there are no more observation nodes. This will require one extra card if there is an exact multiple of 16 observation nodes.

DATASET 22: Element Incidence and Pinch Node Data (one or two cards for each of NE elements)

Variable	Format	Description
Card A: (always required for each element)		
LL	I6	Number of element to which data on this card (and the optional next card) refers. If pinch nodes exist in element LL, then the element number must be specified with a minus sign.
		NODE INCIDENCE LIST
IIN(1)	I6	Number of node 1
IIN(2)	I6	Number of node 2
IIN(3)	I6	Number of node 3
IIN(4)	I6	Number of node 4
		List of corner node numbers in element LL, beginning at any node, but taken in an order counterclockwise about the element.

Card B: (OPTIONAL) - is required immediately following Card A only when LL is negative, O M I T when LL is positive)

		PINCH-NODE INCIDENCE LIST	
IEDGE(1)	I6	Node number of	IIN(1) and IIN(2)
IEDGE(2)	I6	pinch node at	IIN(2) and IIN(3)
IEDGE(3)	I6	mid-point of	IIN(3) and IIN(4)
IEDGE(4)	I6	edge between	IIN(4) and IIN(1)

nodes:

A blank in the list of pinch node numbers indicates that no pinch node exists on that particular edge element LL.

End of Input Data List for UNIT 5

(p.275 follows)

List of Input Data for UNIT 55

Model Series: SUTRA
Model Version: V1284-2D

The data in UNIT 55 need be created by the user only for Cold-Starts of SUTRA simulation (i.e.: for the first time step of a given simulation).

The Restart options are controlled by IREAD and ISTORE in DATASET 4 of UNIT 5 data. SUTRA will optionally store final results of a simulation in a form directly useable as UNIT 55 for later restarts.

DATASET 1: Simulation Starting Time (one card)

Variable	Format	Description
TSTART	G20.0	Elapsed time at which the initial conditions for simulation specified in UNIT 55 are given. [s] This sets the simulation clock starting time. Usually set to a value of zero for Cold-Start.

DATASET 2: Initial Pressure Values at Nodes

Requires (NN + 3)/4 cards. (Done by integer arithmetic.)

Variable	Format	Description
PVEC(II)	4G20.0	Initial (starting) pressure values at time, TSTART, at each of NN nodes. $[M/(L \cdot s^2)]$ Four values per card, in exact order of node numbers. These values are arbitrary and may be left blank if the steady-state flow option in DATASET 4 of UNIT 5 has been chosen. Initial hydrostatic or natural pressures in a cross-section may be obtained by running a single steady-flow time step with the store option. Then the natural pressures are calculated and stored on UNIT 66, and may be copied to the Cold-Start UNIT 55 file without change in format, as initial conditions for a transient run.

DATASET 3: Initial Temperature or Concentration Values at Nodes

Requires (NN+3)/4 cards. (Done by integer arithmetic.)

Variable	Format	Description
UVEC(II)	4G20.0	Initial (starting) temperature or solute concentration (mass fraction) values at time, TSTART, at each of NN nodes. [°C] or [M_s/M] Four values per card, in exact order of node numbers.

End of Input Data List for UNIT 55

(p.279 follows)

REFERENCES

(p. 281 follows)

REFERENCES

Bear, Jacob, 1979, Hydraulics of Groundwater, McGraw-Hill, New York, 567 p.

Desai, C. S., and Contractor, D. N., 1977, Finite element analysis of flow, diffusion, and salt water intrusion in porous media: in Formulation and Computational Algorithms in Finite Element Analysis, by Bathe, K. J., (editor) and others, MIT Press, p. 958-983.

Frind, E. O., 1982, Simulation of long-term transient density-dependent transport in groundwater, Advances in Water Resources, v. 5, p. 73-97.

Gelhar, L. W., and Axness, Carl L., 1983, Three-dimensional stochastic analysis of macrodispersion in aquifers, Water Resources Research, v. 19, no. 1, p. 161-180.

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

Henry, H. R., 1964, Effects of dispersion on salt encroachment in coastal aquifers: in Sea Water in Coastal Aquifers, U.S. Geological Survey Water-Supply Paper 1613-C, p. C71-C84.

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

Huyakorn, P. S., and Pinder, George F., 1983, Computational Methods in Subsurface Flow, Academic Press, New York, 473 p.

Huyakorn, P., and Taylor, C., 1976, Finite element models for coupled groundwater flow and convective dispersion: in Finite Elements in Water Resources by Gray, W.G., Pinder, G. F., and Brebbia, C. A., (editors), Pentech Press, London, 1.131-1.151.

INTERA, 1979, Revision of the documentation for a model for calculating effects of liquid waste disposal in deep saline aquifers, U.S. Geological Survey Water Resources Investigations 79-96, 73 p.

Konikow, L. F., 1977, Modeling chloride movement in the alluvial aquifer at the Rocky Mountain Arsenal, Colorado, U.S. Geological Survey Water Supply Paper 2044, 43 p.

Lohman, S. W., 1972, Ground Water Hydraulics, U.S. Geological Survey Professional Paper 708, 70 p.

Pinder, G. F., and Cooper, H. H., Jr., 1970, A numerical technique for calculating the transient position of the saltwater front, Water Resources Research, v. 6, no. 3, p. 875-882.

Pinder, G. F., and Gray, W. G., 1977, Finite Element Simulation in Surface and Subsurface Hydrology, Academic Press, New York, 295 p.

Segol, G., Pinder, G. F., Gray, W. G., 1975, A galerkin-finite element technique for calculating the transient position of the saltwater front, Water Resources Research, v. 11, no. 2, p. 343-346.

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

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

Wang, H. F., and Anderson, M. P., 1982, Introduction to Groundwater Modeling, Freeman and Co., San Francisco, 237 p.

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

(p. 285 follows)

APPENDICES

(p. 287 follows)

Appendix A

Nomenclature

Generic Units

[1]	unity - implies dimensionless or $[L^0]$
[E]	energy units or $[M \cdot L^2/s^2]$
[L]	length units
$[L_f^3]$	fluid volume
$[L_G^3]$	solid grain volume
[M]	fluid mass units
$[M_G]$	solid grain mass units
$[M_s]$	solute mass units

Units

[°C]	degrees Celcius
[cm]	centimeters
[d]	days
[gr]	grams
[h]	hours
[J]	Joules or $[k_g \cdot m^2/s^2$
[kg]	kilograms mass
[lbm]	pounds mass
[m]	meters
[min]	minutes
[mo]	months
[s]	seconds

Special Notation

$\frac{\partial \Psi}{\partial t}$ or $\frac{d\Psi}{dt}$	time derivative of Ψ
$\underline{v} = \underline{i}\, v_x + \underline{j}\, v_y + \underline{k}\, v_z$	vector $\underline{v}$ with components in $\underline{i}$, $\underline{j}$, and $\underline{k}$ directions
$\underline{\nabla}\Psi = \underline{i}\, \frac{\partial \Psi}{\partial x} + \underline{j}\, \frac{\partial \Psi}{\partial y} + \underline{k}\, \frac{\partial \Psi}{\partial z}$	gradient of scale Ψ
$\underline{\nabla} \cdot \underline{v} = \frac{\partial v}{\partial x} x + \frac{\partial v}{\partial y} y + \frac{\partial v}{\partial z} z$	divergence of vector $\underline{v}$
$i = \overline{1,NN} = 1,2,3,4,\ldots,NN$	index i takes on all integer values between one and NN
$\lvert\Psi\rvert$	absolute value of scalar Ψ
$\lvert\underline{v}\rvert$	magnitude of vector $\underline{v}$
$\hat{\Psi}$ or $\Psi^{\wedge}$	approximate or discretized value of Ψ
$\Delta\Psi$	discrete change in value of Ψ (e.g : $\Delta\Psi = \Psi_1 - \Psi_2$)
Ψ_o	initial condition or zeroth value of Ψ
Ψ_{BC}	value of Ψ as specified at a boundary condition node
Ψ_i or Ψ_j	value of Ψ at node or cell i or j
Ψ_{IN}	value of Ψ in inflow
Ψ_{KG}	value of Ψ at the KG^{th} Gauss point
Ψ_L	value of Ψ in element L
v_s	value of a vector $\underline{v}$ along a stream line
v_x	value of a vector $\underline{v}$ in x direction
v_y	value of a vector $\underline{v}$ in y direction

v_ξ	value of a vector $\underline{v}$ in the ξ direction
v_η	value of a vector $\underline{v}$ in the η direction
Ψ^L	value of Ψ in element L
Ψ^n	value of Ψ at time step n
Ψ^{n+1}	value of Ψ at time step n+1
$\Psi^{(n+1)*}$	value of Ψ evaluated at previous time step on first iteration, and at most recent iteration on subsequent iterations
Ψ^{proj}	value of Ψ projected from previous time steps on first iteration
$\hat{v}^*$	consistently evaluated velocity
$\hat{\rho g}^*$	consistently evaluated density-gravity term
$\sum_{i=1}^{NN} \Psi_i = \Psi_1 + \Psi_2 + \Psi_3 + \cdots + \Psi_{NN}$	summation

Greek Lowercase

α	(2.17)	$[M/(L\cdot s^2)]^{-1}$	Porous matrix compressibility
$\alpha_L(x,y,t)$	(2.40b) (2.41)	[L]	Longitudinal dispersivity of solid matrix
$\alpha_{Lmax}(x,y)$	(2.42b)	[L]	Longitudinal dispersivity in the maximum permeability direction, x_p
$\alpha_{Lmin}(x,y)$	(2.42b)	[L]	Longitudinal dispersivity in the minimum permeability direction, x_m
$\alpha_T(x,y)$	(2.40b)	[L]	Transverse dispersivity of solid matrix
β	(2.15)	$[M/(L\cdot s^2)]^{-1}$	Fluid compressibility
$\gamma_o^s(x,y,t)$	(2.25)	$[E/M_G\cdot s]$	Energy source in solid grains

γ_o^s	(2.37b)	$[(M_s/M)/s]$	Zero-order adsorbate mass production rate
$\gamma_o^w(x,y,t)$	(2.25)	$[E/M\cdot s]$	Energy source in fluid
γ_o^w	(2.37b)	$[(M_s/M)/s]$	Zero-order solute mass production rate
γ_1^s	(2.37b)	$[s^{-1}]$	First-order mass production rate of adsorbate
γ_1^w	(2.37b)	$[s^{-1}]$	First order mass production rate of solute
δ_{ij}	(4.65a)		Kronecker delta
$\varepsilon(x,y,t)$	(2.6)	[1]	Porosity
η	(4.3)		η local coordinate
$\kappa_1(C,C_s)$	(2.32b)	$[M/M_G]$	First general sorption coefficient
$\kappa_2(C,C_s)$	(2.32b)	$[M/M_G\cdot s]$	Second general sorption coefficient
$\kappa_3(C,C_s)$	(2.32b)	$[M_s/M_G\cdot s]$	Third general sorption coefficient
$\lambda(x,y,t)$	(2.25)	$[E/(s\cdot L\cdot{}^\circ C)]$	Bulk thermal conductivity of solid matrix plus fluid
λ_s	(2.26)	$[E/(s\cdot L\cdot{}^\circ C)]$	Solid thermal conductivity (about $\lambda_s \sim 0.6$ $[J/(s\cdot m\cdot{}^\circ C)]$ at 20°C)
λ_w	(2.26)	$[E/(s\cdot L\cdot{}^\circ C)]$	Fluid thermal conductivity (about $\lambda_w \sim 0.6$ $[J/(s\cdot m\cdot{}^\circ C)]$ at 20°C)
μ	(2.5),(2.6)		Fluid viscosity
ν_i	(4.51)		Pressure-based conductance for specified pressure in cell i
ν_p	(4.38)		Conductance for specified pressure nodes

ξ	(4.1)		ξ local coordinate
ρ_o	(2.4)	$\lvert M/L_f^3 \rvert$	Base fluid density at $C=C_o$ or $T=T_o$
$\rho(x,y,t)$	(2.1)	$\lvert M/L_f^3 \rvert$	Fluid density
ρ_s	(2.24) (2.30)	$\lvert M_G^3/L_G^3 \rvert$	Density of solid grains in solid matrix
σ'	(2.17)	$\lvert M/(L\cdot s^2) \rvert$	Integranular stress
σ_s	(2.47)		Diffusion in solid phase in unified transport equation
σ_w	(2.47)		Diffusion in fluid phase in unified transport equation
$\theta(x,y)$	(2.21a)	$\lvert ^\circ \rvert$	Angle from +x-coordinate axis to direction of maximum permeability, x_p
$\phi_{kv}(x,y,t)$	(2.42b)	$\lvert ^\circ \rvert$	Angle from maximum permeability direction, x_p to local flow direction, $(\underline{v}/\lvert\underline{v}\rvert)$
ϕ_j	(3.4)		Symmetric bi-linear basis function in global coordinates at node i
χ_1	(2.34b)	$\lvert L_f^3/M_G \rvert$	Linear distribution coefficient
χ_1	(2.35b)	$\lvert L_f^3/M_G \rvert$	A Freundlich distribution coefficient
χ_1	(2.36b)	$\lvert L_f^3/M_G \rvert$	A Langmuir distribution coefficient
χ_2	(2.36b)	$\lvert L_f^3/M_s \rvert$	Langmuir coefficient
χ_2	(2.35b)	$\lvert 1 \rvert$	Freundlich coefficient
ψ_{IN_i}	(4.75)		Energy source $\lvert E/s \rvert$ or solute mass source $\lvert M_s/M\cdot s \rvert$ at node i
ψ_{OUT_i}	(4.75)		Sink of energy or solute mass at node i
ω_i	(4.43)		Asymmetric weighting function in global coordinates at node i

Greek Uppercase

Γ	(3.20)		External boundary of simulated region
$\Gamma_s(x,y,t)$	(2.30)	$\lvert M_s/M_G \cdot s \rvert$	Adsorbate mass source (per unit solid matrix mass) due to production reactions within adsorbed material itself
$\Gamma_w(x,y,t)$	(2.30)	$\lvert M_s/M \cdot s \rvert$	Solute mass source in fluid (per unit fluid mass) due to production reactions
Δt	(7.1)	$\lvert s \rvert$	Length of time step
ΔL_L	(7.4)		Distance between sides of element L along stream line
ΔL_T	(7.5)		Distance between sides of element L perpendicular to stream line
Δt_n	(3.33)		Time step n
Δt_{n+1}	(3.29)		Time step n+1
H_+	(4.3)		One-dimensional basis function in η direction
H_-	(4.3)		One-dimensional basis function in η direction
H^*	(4.18)		Asymmetric portion of η weighting function
Θ_i	(4.13)		Asymmetric weighting function at node i
Ξ_+	(4.2)		One-dimensional basis function in ξ direction
Ξ_-	(4.1)		One-dimensional basis function in ξ direction
Ξ^*	(4.17)		Asymmetric portion of ξ weighting function

$T(x,y,t)$	(2.22)	$[M/(L^3 \cdot s)]$	Solute mass source (e.g., dissolution of solid matrix or desoprtion)
Ω_i	(4.8)		Bi-linear symmetric basis function at node i

Roman Lowercase

a_ξ	(4.23)		Asymmetric weighting function coefficient
$c(x,y,t)$	(2.1)	$[M_s/L_f^3]$	Solute volumetric concentration (mass solute per volume total fluid)
c_s	(2.27b)	$[E/(M_G \cdot °C)]$	Solid grain specific heat (about $c_s \sim 8.4 \times 10^2 [J/(kg \cdot °C]$ for sandstone at 20°C)
c_w	(2.25)	$[E/(M \cdot °C)]$	Specific heat of water (about $c_w \sim 4.182 \times 10^3 [J/(kg \cdot °C)]$ at 20°C
$d_L(x,y,t)$	(2.39c)	$[L^2/s]$	Longitudinal dispersion coefficient
$d_T(x,y,t)$	(2.39c)	$[L^2/s]$	Transverse dispersion coefficient
det J	(4.30)		Determinant of Jacobian matrix
e_s	(2.24)	$[E/M_G]$	Energy per unit mass solid matrix
e_w	(2.24)	$[E/M]$	Energy per unit mass water
$f(x,y,t)$	(2.30)	$[M_s/(L^3 \cdot s)]$	Volumetric adsorbate source (gain of adsorbed species by transfer from fluid per unit from fluid per unit total volume)
$f_s(x,y,t)$	(2.32a)	$[M_s/M_G \cdot s]$	Specific solute mass adsorption rate (per unit mass solid matrix)
$\underline{g}$	(2.19b)	$[L/s^2]$	Gravitational acceleration (gravity vector)

$h(x,y,t)$	(2.20) (3.1)	$\|L\|$	Hydraulic head (sum of pressure head and elevation head)
$\underline{\underline{k}}(x,y)$	(2.19a)	$\|L^2\|$	Solid matrix permeability
$k_{max}(x,y)($	(2.21a)	$\|L^2\|$	Absolute maximum value of permeability
$k_{min}(x,y)$	(2.21a)	$\|L^2\|$	Absolute minimum value of permeability
$k_r(x,y,t)$	(2.19)	$\|1\|$	Relative permeability to fluid flow (assumed to be independent of direction).
$p(x,y,t)$	(2.1)	$\|M/(L\cdot s^2)\|$	Fluid pressure
$P_c(x,y,t)$	(2.7)	$\|M/(L\cdot s^2)\|$	Capillary pressure
Pcent	(2.7)	$\|M/(L\cdot s^2)\|$	Entry capillary pressure
p_{BC_i}	(4.38)		Specified pressure value at node i
q_{IN_i}	(4.44)		Fluid mass flux in across boundary at node i
q_{OUT_i}	(4.44)		Fluid mass flux out across boundary node i
r^*	(6.3a)		Parameter in analytical solution for radial transport
s_L	(4.84)		Left side coefficient contribution of sorption isotherm to U equation
s_R	(4.84)		Right side contribution of isotherm to U equation
t	(3.4)		Time
$v(x,y,t)$	(2.39c)	$\|L/s\|$	Magnitude of velocity $\underline{v}$
$\underline{v}(x,y,t)$	(2.19a)	$\|L/s\|$	Average fluid velocity
$\underline{v}_s$	(2.49)	$\|L/s\|$	Net solid matrix velocity
$v_x(x,y,t)$	(2.39c)	$\|L/s\|$	Magnitude of x-component of $\underline{v}$
$v_y(x,y,t)$	(2.39c)	$\|L/s\|$	Magnitude of y-component of $\underline{v}$

x		$\|L\|$	x coordinate
x_m			Minor principal direction
x_p			Major principal direction
y		$\|L\|$	y coordinate

Roman Uppercase

A	(6.3b)		Factor in analytical solution for radial transport
AF_i	(4.53)		Matrix coefficient of pressure time derivative
AT_i	(4.86)		Matrix coefficient of U time derivative
$B(x,y,t)$	(3.2)	$\|L\|$	Aquifer thickness
$BASE(x,y)$	(3.2)	$\|L\|$	Elevation of aquifer base for example problem
BF_{ij}	(4.55)		Matrix coefficient in pressure equation
BT_{ij}	(4.88)		Matrix coefficient in U equation
C_o	(2.4)	$\|M_s/M\|$	Base fluid solute concentration
$C(x,y,t)$	(2.1)	$\|M_s/M\|$	Fluid solute mass fraction (or solute concentration) (mass solute per mass total fluid)
$C_s(x,y,t)$	(2.30)	$\|M_s/M_G\|$	Specific concentration of adsorbate on solid grains (mass adsorbate/(mass solid grains plus adsorbate))
$C^*(x,y,t)$	(2.30)	$\|M_s/M\|$	Solute concentration of fluid sources (mass fraction))
CF_i	(4.54)		Matrix coefficient of U time derivative in pressure equation

$\underline{\underline{D}}(x,y,t)$	(2.25),(2.29)	$\lvert L^2/s \rvert$	Dispersion tensor
D_m	(2.29)	$\lvert L^2/s \rvert$	Apparent molecular diffusivity of solute in solution in a porous medium including tortuosity effects, (about, $D_m \sim 1. \times 10$ $\lvert m^2/s \rvert$ for NaCl at 20.°C)
D_{ij}	(2.39c)	$\lvert L^2/s \rvert$	Element of dispersion tensor
D_{xx}	(2.39a)	$\lvert L^2/s \rvert$	Element of dispersion tensor
D_{yy}	(2.39b)	$\lvert L^2/s \rvert$	Element of dispersion tensor
DF_i	(4.56)		Element of vector on right side of pressure equation
DT_{ij}	(4.87)		Matrix coefficient of U equation
ET_i	(4.90)		Element of vector on right side of U equation
F_m	(2.42b)		Dispersive flux in principal direction m
F_p	(2.42a)		Dispersive flux in principal direction p
F_s	(2.41)		Dispersive flux along stream line
G_{KG}	(4.32)		Coefficient of Gauss integration
G_sTL	(4.89b)		Element of vector on left side of U equation
G_sTR	(4.89c)		Element of vector on right side of U equation
GT_i	(4.89a)		Element of vector on left side of U equation
$\underline{\underline{I}}$	(2.25)	$\lvert 1 \rvert$	Identity tensor (ones on diagonal, zeroes elsewhere)
I_{ij}	(3.23)		Matrix arising from integral in example problem

IMVDIM	(7.8)		Program dimension
K(x,y)	(2.20) (3.1)	|L/s|	Hydraulic conductivity
KG	(4.32)		Number of Gauss point
NE	(3.3)		Number of elements in mesh
NN	(3.4)		Number of nodes in mesh
NP	(4.32)		Number of Gauss points
NPBC	(7.1)		Number of specified pressure nodes in mesh
NSOP	(7.1)		Number of specified fluid source nodes in mesh
NSOU	(7.1)		Number of specified U source nodes in mesh
NUBC	(7.1)		Number of specified U nodes in mesh
NPCYC	(7.1)		Pressure solution cycle
NUCYC	(7.1)		U solution cycle
O	(3.7)		The governing equation of the example problem
O_p	(4.38)		The fluid mass balance equation
O_u	(4.66)		The energy or solute mass balance equation
Pe_m	(7.1)		The mesh Peclet number
PBC_{ipu}	(7.1)		The ipu^{th} pressure boundary condition value
Q_i	(4.50)	|M/s|	Total fluid mass source to cell i
$Q_p(x,y,t)$	(2.22)	$\|M/(L^3 \cdot s)\|$	Fluid mass source (including pure water mass plus solute mass dissolved in source water)
$Q^*(x,y)$	(3.1)	$\|s^{-1}\|$	Volumetric fluid source for example problem (volume fluid injected per time /

			example problem (volume fluid injected per time / volume aquifer)
Q_{PBC}	(4.51)	$[M/L^3 \cdot s]$	Fluid mass source rate due to a specified pressure
Q_{BC_i}	(3.38)		Fluid volumetric source due to a specified head in the example problem
Q_{BC_i}	(4.64)	$[M/s]$	Fluid mass source due to a specified pressure node
Q_{IN_i}	(3.20)		Fluid volume efflux at boundary for example problem
Q_{TOT}	(6.1a)		Total pumping rate for pump-test example
Q_i^*	(3.28)		Fluid volumetric source for example problem
R	(3.8)		Residual of discretized equation
RMDIM	(7.6)		Program matrix dimension
RVDIM	(7.7)		Program matrix dimension
$S_{op}(x,y)$	(2.13)	$[M_f/(L \cdot s^2)]^{-1}$	Specific pressure storativity
$S(x,y)$	(3.1)	$[L^{-1}]$	Specific storativity for example problem
S^*	(6.1a)		Dimensionless drawdown for pump test example
$S_w(x,y,t)$	(2.6)	[1]	Water saturation (saturation) (volume of water per volume of voids)
T_o	(2.3)	[°C]	Base fluid temperature
$T(x,y,t)$	(2.1)	[°C]	Fluid temperature (degrees Celcius)
$\mathcal{T}(x,y,t)$	(3.2)	$[L^2/s]$	Aquifer transmissivity for example problem
$T^*(x,y,t)$	(2.25)	[°C]	Temperature of source fluid

U	(2.47)	[°C] or $[M_s/M]$	either T or C depending on type of simulation
U_{BC}	(4.66)		U value of inflow at point of specified pressure
U^*	(2.47a)		U value of fluid source
UP	(4.23)		Upstream weighting factor
V_i	(3.15)		Cell volume at node i
VOL	(2.9)		Volume (total)
VOL_w	(2.13)		Fluid volume
W_o	(4.111b)		Weight for Langmuir isotherm
W(u)	(6.1a)		Well function for pump test example
W_i	(4.39)		Weighting function
W_∞	(4.111a)		Weight for Langmuir isotherm

(p. 301 follows)

Appendix B:

SUTRA Program Listing

(Model version V1284-2D)

(p. 303 follows)

```
C     SUTRA              M A I N   P R O G R A M   SUTRA-VERSION 1284-2D A10.....
C-------------------------------------------------------------------------A20.....
C|                                                                        |A30.....
C|                                                                        |A40.....
C|                UNITED STATES GEOLOGICAL SURVEY                         |A50.....
C|   GROUNDWATER FLOW AND ENERGY OR SOLUTE TRANSPORT SIMULATION MODEL     |A60.....
C|                                                                        |A70.....
C|                                                                        |A80.....
C|                                                                        |A90.....
C|                                                                        |A100....
C|                     ----------------------                             |A110....
C|                     |                    |                             |A120....
C|                     |   S  U  T  R  A    |                             |A130....
C|                     |--------------------|                             |A140....
C|                                                                        |A150....
C|                                                                        |A160....
C|                Saturated    Unsaturated    TRAnsport                   |A170....
C|                =            =              ===                         |A180....
C|                                                                        |A190....
C|                                                                        |A200....
C|                                                                        |A210....
C|     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *    |A220....
C|     * ->saturated and/or unsaturated groundwater flow                *  |A230....
C|     * ->either single species reactive solute transport              *  |A240....
C|     *    or thermal energy transport                                 *  |A250....
C|     * ->two-dimensional areal or cross-sectional simulation         *  |A260....
C|     * ->either cartesian or radial/cylindrical coordinates          *  |A270....
C|     * ->hybrid galerkin-finite-element method and                   *  |A280....
C|     *    integrated-finite-difference method                         *  |A290....
C|     *    with two-dimensional quadrilateral finite elements          *  |A300....
C|     * ->finite-difference time discretization                       *  |A310....
C|     * ->non-linear iterative, sequential or steady-state            *  |A320....
C|     *    solution modes                                              *  |A330....
C|     * ->optional fluid velocity calculation                         *  |A340....
C|     * ->optional observation well output                            *  |A350....
C|     * ->optional printer plots of output                            *  |A360....
C|     * ->optional fluid mass and solute mass or energy budget        *  |A370....
C|     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *    |A380....
C|                                                                        |A390....
C|                                                                        |A400....
C|                                                                        |A410....
C|       Complete explanation of function and use of this code            |A420....
C|       is given in :                                                    |A430....
C|                                                                        |A440....
C|       Voss, Clifford I., 1984, SUTRA: A Finite-Element                 |A450....
C|            Simulation Model for Saturated-Unsaturated                  |A460....
C|            Fluid-Density-Dependent Ground-Water Flow                   |A470....
C|            with Energy Transport or Chemically-Reactive                |A480....
C|            Single-Species Solute Transport, U.S. Geological            |A490....
C|            Survey Water-Resources Investigations Report                |A500....
C|            84-4369.                                                   |A510....
C|                                                                        |A520....
C|                                                                        |A530....
C|                                                                        |A540....
C|       Users who wish to be notified of updates of the SUTRA            |A550....
C|       code and documentation may be added to the mailing               |A560....
C|       by sending a request to :                                        |A570....
C|                                                                        |A580....
C|                     Chief Hydrologist - SUTRA                          |A590....
C|                      U.S. Geological Survey                            |A600....
```

```
C|                          431 National Center                                |A610....
C|                         Reston, Virginia 22092                              |A620....
C|                                 USA                                         |A630....
C|                                                                             |A640....
C|     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *     |A650....
C|     * The SUTRA code and documentation were prepared under a  *             |A660....
C|     * joint research project of the U.S. Geological Survey,  *             |A670....
C|     * Department of the Interior, Reston, Virginia, and the  *             |A680....
C|     * Engineering and Services Laboratory,  U.S. Air Force   *             |A690....
C|     * Engineering and Services Center, Tyndall A.F.B.,       *             |A700....
C|     * Florida.  The SUTRA code and documentation are         *             |A710....
C|     * available for unlimited distribution.                  *             |A720....
C|     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *     |A730....
C|                                                                             |A740....
C|                                                                             |A750....
C|                                                                             |A760....
C|-----------------------------------------------------------------------------|A770....
C                                                                               A780....
C                                                                               A790....
C                                                                               A800....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                                       A810....
      COMMON/LGEM/ RM                                                           A820....
      COMMON/LGEV/ RV                                                           A830....
      COMMON/LGEMV/ IMV                                                         A840....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,                    A850....
     1   NSOP,NSOU,NBCN                                                         A860....
      COMMON/CONTRL/ GNJ,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,                A870....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                          A880....
      COMMON/OBS/ NOBSN,NTOBSN,NOBCYC,ITCNT                                     A890....
      CHARACTER*1 TITLE1(80),TITLE2(80)                                         A900....
      CHARACTER*6 SIMULA(2)                                                     A910....
      DIMENSION KRV(100)                                                        A920....
C                                                                               A930....
C                                                                               A940....
C-------------------------------------------------------------------------      A950....
C|* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *|     A960....
C|* ***************************************************************** *|     A970....
C|* *                                                               * *|     A980....
C|* *   The three arrays that need be dimensioned                   * *|     A990....
C|* *    are dimensioned as follows:                                * *|     A1000...
C|* *                                                               * *|     A1010...
C|* *   DIMENSION  RM( RMDIM), RV( RVDIM), IMV(IMVDIM)              * *|     A1020...
C|* *                                                               * *|     A1030...
C|* *   RMDIM >= 2*NN*NBI                                           * *|     A1040...
C|* *                                                               * *|     A1050...
C|* *   RVDIM >= (( NNV*NN + (NEV+8)*NE + NBCN*3                    * *|     A1060...
C|* *               + (NOBS+1)*(NTOBS+2)*2 + NTOBS + 5 ))           * *|     A1070...
C|* *                                                               * *|     A1080...
C|* *   IMVDIM >= (( NE*8 + NN + NPINCH*3 + NSOP + NSOU             * *|     A1090...
C|* *               + NBCN*2 + NOBS + NTOBS + 12 ))                 * *|     A1100...
C|* *                                                               * *|     A1110...
C|* *   where:                                                      * *|     A1120...
C|* *                                                               * *|     A1130...
C|* *    NNV = 30                                                   * *|     A1140...
C|* *    NEV = 10                                                   * *|     A1150...
C|* *    NBCN = NPBC + NUBC                                         * *|     A1160...
C|* *                                                               * *|     A1170...
C|* *   and:                                                        * *|     A1180...
C|* *                                                               * *|     A1190...
C|* *    NN = number of nodes in finite element mesh                * *|     A1200...
```

```
C     SUTRA              M A I N   P R O G R A M   SUTRA-VERSION 1284-2D A10.....


C|* *     NE = number of elements in finite element mesh              * *|  A1210...
C|* *     NOBS = number of observation nodes in mesh                  * *|  A1220...
C|* *     NTOBS = maximum number of time steps with observations      * *|  A1230...
C|* *     NPINCH = number of pinch nodes in finite element mesh       * *|  A1240...
C|* *     NSOP = number of fluid mass source nodes in mesh            * *|  A1250...
C|* *     NSOU = number of energy or solute mass source nodes         * *|  A1260...
C|* *     NPBC = number of specified pressure nodes in mesh           * *|  A1270...
C|* *     NUBC = number of specified concentration or temperature     * *|  A1280...
C|* *            nodes in mesh                                        * *|  A1290...
C|* *                                                                 * *|  A1300...
C|* *                                                                 * *|  A1310...
C|* *   The three arrays must be given dimensions just below.         * *|  A1320...
C|* *                                                                 * *|  A1330...
C|* ***************************************************************** *|  A1340...
      DIMENSION   RM(040000), RV( 30000), IMV( 10000)                      A1350...
C|* ***************************************************************** *|  A1360...
C|* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *|  A1370...
C|-----------------------------------------------------------------------|  A1380...
C                                                                          A1390...
C                                                                          A1400...
C                                                                          A1410...
C.....INPUT DATASET 1:  INPUT DATA HEADING                                 A1420...
C.....( SET ME=-1 FOR SOLUTE TRANSPORT, ME=+1 FOR ENERGY TRANSPORT )       A1430...
      READ(5,100) SIMULA                                                   A1440...
  100 FORMAT(2A6)                                                          A1450...
      WRITE(6,110)                                                         A1460...
  110 FORMAT(1H1,132(1H*)////3(1X,132(1H*)/////////                        A1470...
     1   47X,' SSSS   UU  UU  TTTTTT  RRRRR     AA  '/                     A1480...
     2   47X,'SS  S  UU  UU  T TT T  RR  RR   AAAA '/                     A1490...
     3   47X,'SSSS    UU  UU    TT    RRRRR   AA  AA'/                    A1500...
     4   47X,'    SS  UU  UU    TT    RR R    AAAAAA'/                    A1510...
     5   47X,'SS  SS  UU  UU    TT    RR RR   AA  AA'/                    A1520...
     6   47X,' SSSS    JUUU     TT    RR  RR  AA  AA'/                    A1530...
     7   7(/),37X,'U N I T E D   S T A T E S  ',                           A1540...
     8   'G E O L O G I C A L   S U R V E Y'////                           A1550...
     9   45X,'SUBSURFACE FLOW AND TRANSPORT SIMULATION MODEL'/             A1560...
     *   //59X,'-VERSION 1284-2D-'///                                     A1570...
     A   36X,'*  SATURATED-UNSATURATED FLOW AND SOLUTE OR ENERGY',        A1580...
     B   ' TRANSPORT  *'////4(////1X,132(1H*)))                           A1590...
C                                                                          A1600...
      IF(SIMULA(1).NE.'SUTRA ') GOTO 115                                   A1610...
      IF(SIMULA(2).EQ.'SOLUTE') GOTO 120                                   A1620...
      IF(SIMULA(2).EQ.'ENERGY') GOTO 140                                   A1630...
  115 WRITE(6,116)                                                         A1640...
  116 FORMAT(1H1/////20X,'* * * * *   ERROR IN FIRST DATA CARD--',        A1650...
     1   '------DATA INPUT HALTED FOR CORRECTIONS   * * * * * *')          A1660...
      STOP                                                                 A1670...
  120 ME=-1                                                                A1680...
      WRITE(6,130)                                                         A1690...
  130 FORMAT(1H1//132(1H*)///20X,'* * * * *   S U T R A   S O L U ',     A1700...
     1   'T E   T R A N S P O R T   S I M U L A T I O N   * * * * *'//     A1710...
     2   /132(1H*)/)                                                      A1720...
      GOTO 160                                                             A1730...
  140 ME=+1                                                                A1740...
      WRITE(6,150)                                                         A1750...
  150 FORMAT(1H1//132(1H*)///20X,'* * * * *   S U T R A   E N E R ',     A1760...
     1   'G Y   T R A N S P O R T   S I M U L A T I O N   * * * * *'//     A1770...
     2   /132(1H*)/)                                                      A1780...
  160 CONTINUE                                                             A1790...
C                                                                          A1800...
```

```
C.....INPUT DATASET 2:  OUTPUT HEADING                                  A1810...
      READ(5,170) TITLE1,TITLE2                                         A1820...
  170 FORMAT(80A1/80A1)                                                 A1830...
      WRITE(6,180) TITLE1,TITLE2                                        A1840...
  180 FORMAT(////1X,131(1H-)//26X,80A1//26X,80A1//1X,131(1H-))          A1850...
      READ(5,200) NN,NE,NBI,NPINCH,NPBC,NUBC,NSOP,NSOU,NOBS,NTOBS      A1860...
      READ(5,200) IUNSAT,ISSFLO,ISSTRA,IREAD,ISTORE                     A1870...
  200 FORMAT(16I5)                                                      A1880...
      WRITE(6,205)                                                      A1890...
  205 FORMAT(/////11X,'S I M U L A T I O N   M O D E   ',               A1900...
     1   'O P T I O N S'/)                                              A1910...
      IF(ISSTRA.EQ.1.AND.ISSFLO.NE.1) THEN                              A1920...
       WRITE(6,210)                                                     A1930...
  210  FORMAT(////11X,'STEADY-STATE TRANSPORT ALSO REQUIRES THAT ',     A1940...
     1    'FLOW IS AT STEADY STATE.'//11X,'PLEASE CORRECT ISSFLO ',     A1950...
     2    'AND ISSTRA IN THE INPUT DATA, AND RERUN.'////////            A1960...
     3    45X,'S I M U L A T I O N   H A L T E D   DUE TO INPUT ERROR') A1970...
       ENDFILE(6)                                                       A1980...
       STOP                                                             A1990...
      ENDIF                                                             A2000...
      IF(IUNSAT.EQ.+1) WRITE(6,215)                                     A2010...
      IF(IUNSAT.EQ.0) WRITE(6,216)                                      A2020...
  215 FORMAT(11X,'- ALLOW UNSATURATED AND SATURATED FLOW:  UNSATURATED',A2030...
     1    ' PROPERTIES ARE USER-PROGRAMMED IN SUBROUTINE   U N S A T')  A2040...
  216 FORMAT(11X,'- ASSUME SATURATED FLOW ONLY')                        A2050...
      IF(ISSFLO.EQ.+1.AND.ME.EQ.-1) WRITE(6,219)                        A2060...
      IF(ISSFLO.EQ.+1.AND.ME.EQ.+1) WRITE(6,220)                        A2070...
      IF(ISSFLO.EQ.0) WRITE(6,221)                                      A2080...
  219 FORMAT(11X,'- ASSUME STEADY-STATE FLOW FIELD CONSISTENT WITH ',   A2090...
     1    'INITIAL CONCENTRATION CONDITIONS')                           A2100...
  220 FORMAT(11X,'- ASSUME STEADY-STATE FLOW FIELD CONSISTENT WITH ',   A2110...
     1    'INITIAL TEMPERATURE CONDITIONS')                             A2120...
  221 FORMAT(11X,'- ALLOW TIME-DEPENDENT FLOW FIELD')                   A2130...
      IF(ISSTRA.EQ.+1) WRITE(6,225)                                     A2140...
      IF(ISSTRA.EQ.0) WRITE(6,226)                                      A2150...
  225 FORMAT(11X,'- ASSUME STEADY-STATE TRANSPORT')                     A2160...
  226 FORMAT(11X,'- ALLOW TIME-DEPENDENT TRANSPORT')                    A2170...
      IF(IREAD.EQ.-1) WRITE(6,230)                                      A2180...
      IF(IREAD.EQ.+1) WRITE(6,231)                                      A2190...
  230 FORMAT(11X,'- WARM START - SIMULATION IS TO BE ',                 A2200...
     1    'CONTINUED FROM PREVIOUSLY-STORED DATA')                      A2210...
  231 FORMAT(11X,'- COLD START - BEGIN NEW SIMULATION')                 A2220...
      IF(ISTORE.EQ.+1) WRITE(6,240)                                     A2230...
      IF(ISTORE.EQ.0) WRITE(6,241)                                      A2240...
  240 FORMAT(11X,'- STORE RESULTS AFTER EACH TIME STEP ON UNIT-66',     A2250...
     1    ' AS BACK-UP AND FOR USE IN A SIMULATION RE-START')           A2260...
  241 FORMAT(11X,'- DO NOT STORE RESULTS FOR USE IN A ',                A2270...
     1    'RE-START OF SIMULATION')                                     A2280...
C                                                                       A2290...
      IF(ME.EQ.-1)                                                      A2300...
     1   WRITE(6,245) NN,NE,NBI,NPINCH,NPBC,NUBC,NSOP,NSOU,NOBS,NTOBS  A2310...
  245 FORMAT(////11X,'S I M U L A T I O N   C O N T R O L   ',          A2320...
     1    'N U M B E R S'//11X,I6,5X,'NUMBER OF NODES IN FINITE-',      A2330...
     2    'ELEMENT MESH'/11X,I6,5X,'NUMBER OF ELEMENTS IN MESH'/        A2340...
     3    11X,I6,5X,'ESTIMATED MAXIMUM FULL BAND WIDTH FOR MESH'//      A2350...
     4    11X,I6,5X,'EXACT NUMBER OF PINCH NODES IN MESH'//             A2360...
     5    11X,I6,5X,'EXACT NUMBER OF NODES IN MESH AT WHICH ',          A2370...
     6    'PRESSURE IS A SPECIFIED CONSTANT OR FUNCTION OF TIME'/       A2380...
     7    11X,I6,5X,'EXACT NUMBER OF NODES IN MESH AT WHICH ',          A2390...
     8    'SOLUTE CONCENTRATION IS A SPECIFIED CONSTANT OR ',           A2400...
```

```
     9   'FUNCTION OF TIME'//11X,I6,5X,'EXACT NUMBER OF NODES AT',      A2410...
     *   ' WHICH FLUID INFLOW OR OUTFLOW IS A SPECIFIED CONSTANT',      A2420...
     A   ' OR FUNCTION OF TIME'/11X,I6,5X,'EXACT NUMBER OF NODES AT',   A2430...
     B   ' WHICH A SOURCE OR SINK OF SOLUTE MASS IS A SPECIFIED ',      A2440...
     C   'CONSTANT OR FUNCTION OF TIME'//11X,I6,5X,'EXACT NUMBER OF ',  A2450...
     D   'NODES AT WHICH PRESSURE AND CONCENTRATION WILL BE OBSERVED',  A2460...
     E   /11X,I6,5X,'MAXIMUM NUMBER OF TIME STEPS ON WHICH ',          A2470...
     F   'OBSERVATIONS WILL BE MADE')                                  A2480...
C                                                                      A2490...
      IF(ME.EQ.+1)                                                     A2500...
     1    WRITE(6,255) NN,NE,NBI,NPINCH,NPBC,NUBC,NSOP,NSOU,NOBS,NTOBS A2510...
  255 FORMAT(////11X,'S I M U L A T I O N   C O N T R O L   ',         A2520...
     1   'N U M B E R S'//11X,I6,5X,'NUMBER OF NODES IN FINITE-',       A2530...
     2   'ELEMENT MESH'/11X,I6,5X,'NUMBER OF ELEMENTS IN MESH'/         A2540...
     3   11X,I6,5X,'ESTIMATED MAXIMUM FULL BAND WIDTH FOR MESH'//       A2550...
     4   11X,I6,5X,'EXACT NUMBER OF PINCH NODES IN MESH'//              A2560...
     5   11X,I6,5X,'EXACT NUMBER OF NODES IN MESH AT WHICH ',           A2570...
     6   'PRESSURE IS A SPECIFIED CONSTANT OR FUNCTION OF TIME'/        A2580...
     7   11X,I6,5X,'EXACT NUMBER OF NODES IN MESH AT WHICH ',           A2590...
     8   'TEMPERATURE IS A SPECIFIED CONSTANT OR ',                     A2600...
     9   'FUNCTION OF TIME'//11X,I6,5X,'EXACT NUMBER OF NODES AT',      A2610...
     *   ' WHICH FLUID INFLOW OR OUTFLOW IS A SPECIFIED CONSTANT',      A2620...
     A   ' OR FUNCTION OF TIME'/11X,I6,5X,'EXACT NUMBER OF NODES AT',   A2630...
     B   ' WHICH A SOURCE OR SINK OF ENERGY IS A SPECIFIED CONSTANT',   A2640...
     C   ' OR FUNCTION OF TIME'//11X,I6,5X,'EXACT NUMBER OF NODES ',    A2650...
     D   'AT WHICH PRESSURE AND TEMPERATURE WILL BE OBSERVED'           A2660...
     E   /11X,I6,5X,'MAXIMUM NUMBER OF TIME STEPS ON WHICH ',          A2670...
     F   'OBSERVATIONS WILL BE MADE')                                  A2680...
C                                                                      A2690...
C                                                                      A2700...
C.....CALCULATE DIMENSIONS FOR POINTERS                                A2710...
C                                                                      A2720...
      NBCN=NPBC+NUBC+1                                                 A2730...
      NSOP=NSOP+1                                                      A2740...
      NSOU=NSOU+1                                                      A2750...
      NPINCH=NPINCH+1                                                  A2760...
      MATDIM=NN*NBI                                                    A2770...
      NIN=NE*8                                                         A2780...
      NOBSN=NOBS+1                                                     A2790...
      NTOBSN=NTOBS+2                                                   A2800...
      MATOBS=NOBSN*NTOBSN                                              A2810...
      NE4=NE*4                                                         A2820...
C                                                                      A2830...
C                                                                      A2840...
C.....SET UP POINTERS FOR REAL MATRICES                                A2850...
C                                                                      A2860...
      KRM1=1                                                           A2870...
      KRM2=KRM1+   MATDIM                                              A2880...
C     KRM3=KRM2+   MATDIM                                              A2890...
C     NOTE: THE LAST POINTER IN THE ABOVE LIST, CURRENTLY, KRM3,       A2900...
C           MAY  N E V E R  BE PASSED TO SUTRA.  IT POINTS TO THE      A2910...
C           STARTING ELEMENT OF THE NEXT NEW MATRIX TO BE ADDED.       A2920...
C           PRESENTLY, SPACE IS ALLOCATED FOR (2) MATRICES.            A2930...
C                                                                      A2940...
C                                                                      A2950...
C.....SET UP POINTERS FOR REAL VECTORS                                 A2960...
C                                                                      A2970...
C     NNV IS NUMBER OF REAL VECTORS THAT ARE NN LONG                   A2980...
      NNV=30                                                           A2990...
C     NEV IS NUMBER OF REAL VECTORS THAT ARE NE LONG                   A3000...
```

```
C     SUTRA              M A I N   P R O G R A M   SUTRA-VERSION 1234-2D A10.....


      NEV=10                                                                A3010...
C                                                                           A3020...
      M2=1                                                                  A3030...
      KRV(1)=1                                                              A3040...
      M1=M2+1                                                               A3050...
      M2=M2+        ( NNV )                                                 A3060...
      DO 400 J=M1,M2                                                        A3070...
  400 KRV(J)=KRV(J-1)+  NN                                                  A3080...
      M1=M2+1                                                               A3090...
      M2=M2+        ( NEV )                                                 A3100...
      DO 410 J=M1,M2                                                        A3110...
  410 KRV(J)=KRV(J-1)+  NE                                                  A3120...
      M1=M2+1                                                               A3130...
      M2=M2+        ( 3 )                                                   A3140...
      DO 420 J=M1,M2                                                        A3150...
  420 KRV(J)=KRV(J-1)+  NBCN                                                A3160...
      M1=M2+1                                                               A3170...
      M2=M2+        ( 2 )                                                   A3180...
      DO 430 J=M1,M2                                                        A3190...
  430 KRV(J)=KRV(J-1)+  MATOBS                                              A3200...
      M2=M2+        ( 1 )                                                   A3210...
      KRV(M2)=KRV(M2-1)+NTOBSN                                              A3220...
      M1=M2+1                                                               A3230...
      M2=M2+        ( 2 )                                                   A3240...
      DO 440 J=M1,M2                                                        A3250...
  440 KRV(J)=KRV(J-1)+  NE4                                                 A3260...
C     NOTE: THE LAST POINTER IN THE ABOVE LIST, CURRENTLY, KRV(J=49),       A3270...
C           MAY  N E V E R  BE PASSED TO SUTRA.  IT POINTS TO THE           A3280...
C           STARTING ELEMENT OF THE NEXT NEW REAL VECTOR TO BE ADDED.       A3290...
C           PRESENTLY, SPACE IS ALLOCATED FOR (48) VECTORS.                 A3300...
C                                                                           A3310...
C                                                                           A3320...
C.....SET UP POINTERS FOR INTEGER VECTORS                                   A3330...
C                                                                           A3340...
      KIMV1=1                                                               A3350...
      KIMV2=KIMV1+     NIN                                                  A3360...
      KIMV3=KIMV2+     NPINCH*3                                             A3370...
      KIMV4=KIMV3+     NSOP                                                 A3380...
      KIMV5=KIMV4+     NSOU                                                 A3390...
      KIMV6=KIMV5+     NBCN                                                 A3400...
      KIMV7=KIMV6+     NBCN                                                 A3410...
      KIMV8=KIMV7+     NN                                                   A3420...
      KIMV9=KIMV8+     NOBSN                                                A3430...
C     KIMV10=KIMV9+     NTOBSN                                              A3440...
C     NOTE: THE LAST POINTER IN THE ABOVE LIST, CURRENTLY, KIMV10,          A3450...
C           MAY  N E V E R  BE PASSED TO SUTRA.  IT POINTS TO THE           A3460...
C           STARTING ELEMENT OF THE NEXT NEW INTEGER VECTOR TO BE ADDED.    A3470...
C           PRESENTLY, SPACE IS ALLOCATED FOR (8) INTEGER VECTORS.          A3480...
C                                                                           A3490...
C                                                                           A3500...
C.....PASS POINTERS TO MAIN CONTROL ROUTINE, SUTRA                          A3510...
      CALL SUTRA( RM(KRM1),RM(KRM2),                                        A3520...
     1   RV(KRV(1)),RV(KRV(2)),RV(KRV(3)),RV(KRV(4)),RV(KRV(5)),           A3530...
     2   RV(KRV(6)),RV(KRV(7)),RV(KRV(8)),RV(KRV(9)),RV(KRV(10)),          A3540...
     3   RV(KRV(11)),RV(KRV(12)),RV(KRV(13)),RV(KRV(14)),RV(KRV(15)),      A3550...
     4   RV(KRV(16)),RV(KRV(17)),RV(KRV(18)),RV(KRV(19)),RV(KRV(20)),      A3560...
     5   RV(KRV(21)),RV(KRV(22)),RV(KRV(23)),RV(KRV(24)),RV(KRV(25)),      A3570...
     6   RV(KRV(26)),RV(KRV(27)),RV(KRV(28)),RV(KRV(29)),RV(KRV(30)),      A3580...
     7   RV(KRV(31)),RV(KRV(32)),RV(KRV(33)),RV(KRV(34)),RV(KRV(35)),      A3590...
     8   RV(KRV(36)),RV(KRV(37)),RV(KRV(38)),RV(KRV(39)),RV(KRV(40)),      A3600...
```

```
C     SUTRA              M A I N   P R O G R A M   SUTRA-VERSION 1284-2D A10.....


     9   RV(KRV(41)),RV(KRV(42)),RV(KRV(43)),RV(KRV(44)),RV(KRV(45)),    A3610...
     *   RV(KRV(46)),RV(KRV(47)),RV(KRV(48)),                            A3620...
     1   IMV(KIMV1),IMV(KIMV2),IMV(KIMV3),IMV(KIMV4),IMV(KIMV5),         A3630...
     2   IMV(KIMV6),IMV(KIMV7),IMV(KIMV8),IMV(KIMV9) )                   A3640...
C                                                                        A3650...
C                                                                        A3660...
      ENDFILE(6)                                                         A3670...
      STOP                                                               A3680...
      END                                                                A3690...
```

```
C     SUBROUTINE        S  U  T  R  A          SUTRA - VERSION 1284-2D B10.....


C     SUBROUTINE        S  U  T  R  A          SUTRA - VERSION 1284-2D B10.....
C                                                                     B20.....
C *** PURPOSE :                                                       B30.....
C ***  MAIN CONTROL ROUTINE FOR SUTRA SIMULATION.                     B40.....
C ***  ORGANIZES DATA INPUT, INITIALIZATION, CALCULATIONS FOR         B50.....
C ***  EACH TIME STEP AND ITERATION, AND VARIOUS OUTPUTS.             B60.....
C ***  CALLS MOST OTHER SUBROUTINES.                                  B70.....
C                                                                     B80.....
      SUBROUTINE SUTRA( PMAT,UMAT,                                    B90.....
     1   PITER,UITER,PM1,UM1,UM2,PVEL,SL,SR,                          B100....
     2   X,Y,THICK,VOL,POR,CS1,CS2,CS3,SW,DSWDP,RHO,SOP,              B110....
     3   QIN,UIN,QUIN,PVEC,UVEC,RCIT,RCITM1,CC,XX,YY,                 B120....
     4   ALMAX,ALMIN,ATAVG,VMAG,VANG,                                 B130....
     5   PERMXX,PERMXY,PERMYX,PERMYY,PANGLE,                          B140....
     6   PBC,UBC,QPLITR,POBS,UOBS,OBSTIM,GXSI,GETA,                   B150....
     7   IN,IPINCH,IQSOP,IQSOU,IPBC,IUBC,INDEX,IOBS,ITOBS )           B160....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                             B170....
      CHARACTER*10 ADSMOD                                             B180....
      COMMON/MODSOR/ ADSMOD                                           B190....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,          B200....
     1   NSOP,NSOU,NBCN                                               B210....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,      B220....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                      B230....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,      B240....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                B250....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,B260....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2     B270....
      COMMON/ITERAT/ RPM,RPMAX,RUM,RUMAX,ITER,ITRMAX,IPWORS,IUWORS    B280....
      COMMON/KPRINT/ KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG    B290....
      COMMON/OBS/ NOBSN,NTOBSN,NOBCYC,ITCNT                           B300....
      DIMENSION QIN(NN),UIN(NN),IQSOP(NSOP),QUIN(NN),IQSOU(NSOU)      B310....
      DIMENSION IPBC(NBCN),PBC(NBCN),IUBC(NBCN),UBC(NBCN),QPLITR(NBCN)B320....
      DIMENSION IN(NIN),IPINCH(NPINCH,3)                              B330....
      DIMENSION X(NN),Y(NN),THICK(NN),SW(NN),DSWDP(NN),RHO(NN),SOP(NN),B340....
     1   POR(NN),PVEL(NN)                                             B350....
      DIMENSION PERMXX(NE),PERMXY(NE),PERMYX(NE),PERMYY(NE),PANGLE(NE),B360....
     1   ALMAX(NE),ALMIN(NE),ATAVG(NE),VMAG(NE),VANG(NE),             B370....
     2   GXSI(NE,4),GETA(NE,4)                                        B380....
      DIMENSION VOL(NN),PMAT(NN,NBI),PVEC(NN),UMAT(NN,NBI),UVEC(NN)   B390....
      DIMENSION PM1(NN),UM1(NN),UM2(NN),PITER(NN),UITER(NN),          B400....
     1   RCIT(NN),RCITM1(NN),CS1(NN),CS2(NN),CS3(NN)                  B410....
      DIMENSION CC(NN),INDEX(NN),XX(NN),YY(NN)                        B420....
      DIMENSION POBS(NOBSN,NTOBSN),UOBS(NOBSN,NTOBSN),OBSTIM(NTOBSN), B430....
     1   IOBS(NOBSN),ITOBS(NTOBSN)                                    B440....
      DATA IT/0/                                                      B450....
C                                                                     B460....
C                                                                     B470....
C                                                                     B480....
C.....INPUT SIMULATION DATA FROM UNIT=5 (DATASETS 3 THROUGH 15B)      B490....
      CALL INDAT1(X,Y,THICK,POR,ALMAX,ALMIN,ATAVG,PERMXX,PERMXY,      B500....
     1   PERMYX,PERMYY,PANGLE,SOP)                                    B510....
C                                                                     B520....
C.....PLOT MESH (INPUT DATASET 16)                                    B530....
      IF(KPLOTP+KPLOTU.GT.0) CALL PLOT(0,1,X,Y,CC,INDEX,XX,YY,PVEC)   B540....
C                                                                     B550....
C.....INPUT FLUID MASS, AND ENERGY OR SOLUTE MASS SOURCES             B560....
C        (DATASETS 17 AND 18)                                         B570....
      CALL ZERO(QIN,NN,0.0D0)                                         B580....
      CALL ZERO(UIN,NN,0.0D0)                                         B590....
      CALL ZERO(QUIN,NN,0.0D0)                                        B600....
```

```
C     SUBROUTINE        S  U  T  R  A          SUTRA - VERSION 1284-2D B10.....


      IF(NSOP-1.GT.0.OR.NSOU-1.GT.0)                                    B610....
     1    CALL SOURCE(QIN,UIN,IQSOP,QUIN,IQSOU,IQSOPT,IQSOUT)           B620....
C                                                                       B630....
C.....INPUT SPECIFIED P AND U BOUNDARY CONDITIONS (DATASETS 19 AND 20)  B640....
      IF(NBCN-1.GT.0) CALL BOUND(IPBC,PBC,IUBC,UBC,IPBCT,IUBCT)         B650....
C                                                                       B660....
C.....SET FLAG FOR TIME-DEPENDENT SOURCES OR BOUNDARY CONDITIONS.       B670....
C     WHEN IBCT=+4, THERE ARE NO TIME-DEPENDENT SPECIFICATIONS.         B680....
      IBCT=IQSOPT+IQSOUT+IPBCT+IUBCT                                    B690....
C                                                                       B700....
C.....INPUT OBSERVATION NODE DATA (DATASET 21)                          B710....
      IF(NOBSN-1.GT.0) CALL OBSERV(0,IOBS,ITOBS,POBS,UOBS,OBSTIM,       B720....
     1   PVEC,UVEC,ISTOP)                                               B730....
C                                                                       B740....
C.....INPUT MESH CONNECTION DATA (DATASET 22)                           B750....
      CALL CONNEC(IN,IPINCH)                                            B760....
C                                                                       B770....
C.....CALCULATE AND CHECK BAND WIDTH                                    B780....
      CALL BANWID(IN)                                                   B790....
C                                                                       B800....
C.....CHECK THAT PINCH NODES HAVE NO SOURCES OR BOUNDARY CONDITIONS     B810....
      IF(NPINCH-1.GT.0) CALL NCHECK(IPINCH,IQSOP,IQSOU,IPBC,IUBC)       B820....
C                                                                       B830....
C.....INPUT INITIAL OR RESTART CONDITIONS AND INITIALIZE PARAMETERS     B840....
C        (READ UNIT-55 DATA)                                            B850....
      CALL INDAT2(PVEC,UVEC,PM1,UM1,UM2,CS1,CS2,CS3,SL,SR,RCIT,SW,DSWDP,B860....
     1   PBC,IPBC,IPBCT)                                                B870....
C                                                                       B880....
C.....SET STARTING TIME OF SIMULATION CLOCK                             B890....
C     TSEC=TSTART                                                       B900....
      TSECPO=TSEC                                                       B910....
      TSECUO=TSEC                                                       B920....
      TMIN=TSEC/60.D0                                                   B930....
      THOUR=TMIN/60.D0                                                  B940....
      TDAY=THOUR/24.D0                                                  B950....
      TWEEK=TDAY/7.D0                                                   B960....
      TMONTH=TDAY/30.4375D0                                             B970....
      TYEAR=TDAY/365.25D0                                               B980....
C                                                                       B990....
C.....OUTPUT INITIAL CONDITIONS OR STARTING CONDITIONS                  B1000...
      IF(ISSTRA.NE.1) CALL PRISOL(0,0,0,PVEC,UVEC,VMAG,VANG,SW)         B1010...
C                                                                       B1020...
C.....SET SWITCHES AND PARAMETERS FOR SOLUTION WITH STEADY-STATE FLOW   B1030...
      IF(ISSFLO.NE.1) GOTO 1000                                         B1040...
      ML=1                                                              B1050...
      NOUMAT=0                                                          B1060...
      ISSFLO=2                                                          B1070...
      ITER=0                                                            B1080...
      DLTPM1=DELTP                                                      B1090...
      DLTUM1=DELTU                                                      B1100...
      BDELP=0.0D0                                                       B1110...
      BDELU=0.0D0                                                       B1120...
      GOTO 1100                                                         B1130...
C                                                                       B1140...
C                                                                       B1150...
C ***********************************************************************B1160...
C.....BEGIN TIME STEP ***************************************************B1170...
C ***********************************************************************B1180...
 1000 IT=IT+1                                                           B1190...
      ITER=0                                                            B1200...
```

```
      ML=0                                                              B1210...
      NOUMAT=0                                                          B1220...
C.....SET NOUMAT TO OBTAIN U SOLUTION BY SIMPLE BACK SUBSTITUTION       B1230...
C        BEGINNING ON SECOND TIME STEP AFTER A PRESSURE SOLUTION         B1240...
C        IF THE SOLUTION IS NON-ITERATIVE (ITRMAX=1)                     B1250...
      IF(MOD(IT-1,NPCYC).NE.0.AND.MOD(IT,NPCYC).NE.0.AND.IT.GT.2        B1260...
     1   .AND.ITRMAX.EQ.1) NOUMAT=1                                     B1270...
C.....CHOOSE SOLUTION VARIABLE ON THIS TIME STEP:                       B1280...
C        ML=0 FOR P AND U, ML=1 FOR P ONLY, AND ML=2 FOR U ONLY.         B1290...
      IF(IT.EQ.1.AND.ISSFLO.NE.2) GOTO 1005                             B1300...
      IF(MOD(IT,NPCYC).NE.0) ML=2                                       B1310...
      IF(MOD(IT,NUCYC).NE.0) ML=1                                       B1320...
C.....MULTIPLY TIME STEP SIZE BY DTMULT EACH ITCYC TIME STEPS           B1330...
      IF(MOD(IT,ITCYC).EQ.0.AND.IT.GT.1) DELT=DELT*DTMULT               B1340...
C.....SET TIME STEP SIZE TO MAXIMUM ALLOWED SIZE, DTMAX                 B1350...
      IF(DELT.GT.DTMAX) DELT=DTMAX                                      B1360...
C.....INCREMENT SIMULATION CLOCK, TSEC, TO END OF NEW TIME STEP         B1370...
 1005 TSEC=TSEC+DELT                                                    B1380...
      TMIN=TSEC/60.D0                                                   B1390...
      THOUR=TMIN/60.D0                                                  B1400...
      TDAY=THOUR/24.D0                                                  B1410...
      TWEEK=TDAY/7.D0                                                   B1420...
      TMONTH=TDAY/30.4375D0                                             B1430...
      TYEAR=TDAY/365.25D0                                               B1440...
C                                                                       B1450...
C.....SET TIME STEP FOR P AND/OR U, WHICHEVER ARE SOLVED FOR            B1460...
C        ON THIS TIME STEP                                              B1470...
      IF(ML-1) 1010,1020,1030                                           B1480...
 1010 DLTUM1=DELTU                                                      B1490...
      DLTPM1=DELTP                                                      B1500...
      GOTO 1040                                                         B1510...
 1020 DLTPM1=DELTP                                                      B1520...
      GOTO 1040                                                         B1530...
 1030 DLTUM1=DELTU                                                      B1540...
 1040 CONTINUE                                                          B1550...
      DELTP=TSEC-TSECP0                                                 B1560...
      DELTU=TSEC-TSECU0                                                 B1570...
C.....SET PROJECTION FACTORS USED ON FIRST ITERATION TO EXTRAPOLATE      B1580...
C        AHEAD ONE-HALF TIME STEP                                       B1590...
      BDELP=(DELTP/DLTPM1)*0.5D0                                        B1600...
      BDELU=(DELTU/DLTUM1)*0.5D0                                        B1610...
      BDELP1=BDELP+1.0D0                                                B1620...
      BDELU1=BDELU+1.0D0                                                B1630...
C.....INCREMENT CLOCK FOR WHICHEVER OF P AND U WILL BE SOLVED FOR       B1640...
C        ON THIS TIME STEP                                              B1650...
      IF(ML-1) 1060,1070,1080                                           B1660...
 1060 TSECP0=TSEC                                                       B1670...
      TSECU0=TSEC                                                       B1680...
      GOTO 1090                                                         B1690...
 1070 TSECP0=TSEC                                                       B1700...
      GOTO 1090                                                         B1710...
 1080 TSECU0=TSEC                                                       B1720...
 1090 CONTINUE                                                          B1730...
C                                                                       B1740...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - B1750...
C.....BEGIN ITERATION - - - - - - - - - - - - - - - - - - - - - - - - - B1760...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - B1770...
 1100 ITER=ITER+1                                                       B1780...
C                                                                       B1790...
      IF(ML-1) 2000,2200,2400                                           B1800...
```

```
C     SUBROUTINE        S  J  T  R  A          SUTRA - VERSION 1284-2D B10.....


C.....SHIFT AND SET VECTORS FOR TIME STEP WITH BOTH P AND J SOLUTIONS   B1810...
 2000 DO 2025 I=1,NN                                                   B1820...
      PITER(I)=PVEC(I)                                                 B1830...
      PVEL(I)=PVEC(I)                                                  B1840...
      UITER(I)=UVEC(I)                                                 B1850...
      RCITM1(I)=RCIT(I)                                                B1860...
 2025 RCIT(I)=RHOWO+DRWDU*(JITER(I)-URHOWO)                            B1870...
      DO 2050 IP=1,NPBC                                                B1880...
      I=IABS(IPBC(IP))                                                 B1890...
      QPLITR(IP)=GNU*(PBC(IP)-PITER(I))                                B1900...
 2050 CONTINUE                                                         B1910...
      IF(ITER.GT.1) GOTO 2600                                          B1920...
      DO 2075 I=1,NN                                                   B1930...
      PITER(I)=BDELP1*PVEC(I)-BDELP*PM1(I)                             B1940...
      UITER(I)=BDELU1*UVEC(I)-BDELU*UM1(I)                             B1950...
      PM1(I)=PVEC(I)                                                   B1960...
      UM2(I)=UM1(I)                                                    B1970...
 2075 UM1(I)=UVEC(I)                                                   B1980...
      GOTO 2600                                                        B1990...
C.....SHIFT AND SET VECTORS FOR TIME STEP WITH P SOLUTION ONLY         B2000...
 2200 DO 2225 I=1,NN                                                   B2010...
      PVEL(I)=PVEC(I)                                                  B2020...
 2225 PITER(I)=PVEC(I)                                                 B2030...
      IF(ITER.GT.1) GOTO 2600                                          B2040...
      DO 2250 I=1,NN                                                   B2050...
      PITER(I)=BDELP1*PVEC(I)-BDELP*PM1(I)                             B2060...
      UITER(I)=UVEC(I)                                                 B2070...
      RCITM1(I)=RCIT(I)                                                B2080...
      RCIT(I)=RHOWO+DRWDU*(JITER(I)-URHOWO)                            B2090...
 2250 PM1(I)=PVEC(I)                                                   B2100...
      GOTO 2600                                                        B2110...
C.....SHIFT AND SET VECTORS FOR TIME STEP WITH U SOLUTION ONLY         B2120...
 2400 IF(NOUMAT.EQ.1) GOTO 2480                                        B2130...
      DO 2425 I=1,NN                                                   B2140...
 2425 UITER(I)=UVEC(I)                                                 B2150...
      IF(ITER.GT.1) GOTO 2600                                          B2160...
      DO 2450 I=1,NN                                                   B2170...
      PITER(I)=PVEC(I)                                                 B2180...
      PVEL(I)=PVEC(I)                                                  B2190...
      UITER(I)=BDELU1*UVEC(I)-BDELU*UM1(I)                             B2200...
 2450 RCITM1(I)=RCIT(I)                                                B2210...
      DO 2475 IP=1,NPBC                                                B2220...
      I=IABS(IPBC(IP))                                                 B2230...
      QPLITR(IP)=GNU*(PBC(IP)-PITER(I))                                B2240...
 2475 CONTINUE                                                         B2250...
 2480 DO 2500 I=1,NN                                                   B2260...
      UM2(I)=UM1(I)                                                    B2270...
 2500 UM1(I)=UVEC(I)                                                   B2280...
 2600 CONTINUE                                                         B2290...
C                                                                      B2300...
C.....INITIALIZE ARRAYS WITH VALUE OF ZERO                             B2310...
      MATDIM=NN*NBI                                                    B2320...
      IF(ML-1) 3000,3000,3300                                          B2330...
 3000 CALL ZERO(PMAT,MATDIM,0.0D0)                                     B2340...
      CALL ZERO(PVEC,NN,0.0D0)                                         B2350...
      CALL ZERO(VOL,NN,0.0D0)                                          B2360...
      IF(ML-1) 3300,3400,3300                                          B2370...
 3300 IF(NOUMAT) 3350,3350,3375                                        B2380...
 3350 CALL ZERO(UMAT,MATDIM,0.0D0)                                     B2390...
 3375 CALL ZERO(UVEC,NN,0.0D0)                                         B2400...
```

```
 3400 CONTINUE                                                          B2410...
C                                                                       B2420...
C.....SET TIME-DEPENDENT BOUNDARY CONDITIONS, SOURCES AND SINKS         B2430...
C         FOR THIS TIME STEP                                            B2440...
      IF(ITER.EQ.1.AND.IBCT.NE.4)                                       B2450...
     1   CALL BCTIME(IPBC,PBC,IUBC,UBC,QIN,UIN,QUIN,IQSOP,IQSOU,        B2460...
     2   IPBCT,IUBCT,IQSOPT,IQSOUT)                                     B2470...
C                                                                       B2480...
C.....SET SORPTION PARAMETERS FOR THIS TIME STEP                        B2490...
      IF(ML.NE.1.AND.ME.EQ.-1.AND.NOUMAT.EQ.0.AND.                      B2500...
     1   ADSMOD.NE.'NONE      ') CALL ADSORB(CS1,CS2,CS3,SL,SR,UITER)   B2510...
C                                                                       B2520...
C.....DO ELEMENTWISE CALCULATIONS IN MATRIX EQUATION FOR P AND/OR U     B2530...
      IF(NOUMAT.EQ.0)                                                   B2540...
     1 CALL ELEMEN(ML,IN,X,Y,THICK,PITER,UITER,RCIT,RCITM1,POR,         B2550...
     2   ALMAX,ALMIN,ATAVG,PERMXX,PERMXY,PERMYX,PERMYY,PANGLE,          B2560...
     3   VMAG,VANG,VOL,PMAT,PVEC,UMAT,UVEC,GXSI,GETA,PVEL)              B2570...
C                                                                       B2580...
C.....DO NODEWISE CALCULATIONS IN MATRIX EQUATION FOR P AND/OR U        B2590...
      CALL NODALB(ML,VOL,PMAT,PVEC,UMAT,UVEC,PITER,UITER,PM1,UM1,UM2,   B2600...
     1   POR,QIN,UIN,QUIN,CS1,CS2,CS3,SL,SR,SW,DSWDP,RHO,SOP)           B2610...
C                                                                       B2620...
C.....SET SPECIFIED P AND U CONDITIONS IN MATRIX EQUATION FOR P AND/OR UB2630...
      CALL BCB(ML,PMAT,PVEC,UMAT,UVEC,IPBC,PBC,IUBC,UBC,QPLITR)         B2640...
C                                                                       B2650...
C.....SET PINCH NODE CONDITIONS IN MATRIX EQUATION FOR P AND/OR U       B2660...
      IF(NPINCH-1) 4200,4200,4000                                       B2670...
 4000 CALL PINCHB(ML,IPINCH,PMAT,PVEC,UMAT,UVEC)                        B2680...
 4200 CONTINUE                                                          B2690...
C                                                                       B2700...
C.....MATRIX EQUATION FOR P AND/OR U ARE COMPLETE, SOLVE EQUATIONS:     B2710...
C         WHEN KKK=0, DECOMPOSE AND BACK-SUBSTITUTE,                    B2720...
C         WHEN KKK=2, BACK-SUBSTITUTE ONLY.                             B2730...
      IHALFB=NBHALF-1                                                   B2740...
      IF(ML-1) 5000,5000,5500                                           B2750...
C.....SOLVE FOR P                                                       B2760...
 5000 KKK=000000                                                        B2770...
      CALL SOLVEB(KKK,PMAT,PVEC,NN,IHALFB,NN,NBI)                       B2780...
C.....P SOLUTION NOW IN PVEC                                            B2790...
      IF(ML-1) 5500,6000,5500                                           B2800...
C.....SOLVE FOR U                                                       B2810...
 5500 KKK=000000                                                        B2820...
      IF(NOUMAT) 5700,5700,5600                                         B2830...
 5600 KKK=2                                                             B2840...
 5700 CALL SOLVEB(KKK,UMAT,UVEC,NN,IHALFB,NN,NBI)                       B2850...
C.....U SOLUTION NOW IN UVEC                                            B2860...
 6000 CONTINUE                                                          B2870...
C                                                                       B2880...
C.....CHECK PROGRESS AND CONVERGENCE OF ITERATIONS                      B2890...
C         AND SET STOP AND GO FLAGS:                                    B2900...
C            ISTOP = -1   NOT CONVERGED - STOP SIMULATION               B2910...
C            ISTOP =  0   ITERATIONS LEFT OR CONVERGED - KEEP SIMULATING B2920...
C            ISTOP =  1   LAST TIME STEP REACHED - STOP SIMULATION      B2930...
C            ISTOP =  2   MAXIMUM TIME REACHED - STOP SIMULATION        B2940...
C            IGOI = 0   P AND U CONVERGED, OR NO ITERATIONS DONE        B2950...
C            IGOI = 1   ONLY P HAS NOT YET CONVERGED TO CRITERION       B2960...
C            IGOI = 2   ONLY U HAS NOT YET CONVERGED TO CRITERION       B2970...
C            IGOI = 3   BOTH P AND U HAVE NOT YET CONVERGED TO CRITERIA B2980...
      ISTOP=0                                                           B2990...
      IGOI=0                                                            B3000...
```

```
C     SUBROUTINE        S  U  T  R  A          SUTRA - VERSION 1284-2D B10.....


      IF(ITRMAX-1) 7500,7500,7000                                          B3010...
 7000 RPM=0.D0                                                             B3020...
      RUM=0.D0                                                             B3030...
      IPWORS=0                                                             B3040...
      IUWORS=0                                                             B3050...
      IF(ML-1) 7050,7050,7150                                              B3060...
 7050 DO 7100 I=1,NN                                                       B3070...
      RP=DABS(PVEC(I)-PITER(I))                                            B3080...
      IF(RP-RPM) 7100,7060,7060                                            B3090...
 7060 RPM=RP                                                               B3100...
      IPWORS=I                                                             B3110...
 7100 CONTINUE                                                             B3120...
      IF(RPM.GT.RPMAX) IGOI=IGOI+1                                         B3130...
 7150 IF(ML-1) 7200,7350,7200                                              B3140...
 7200 DO 7300 I=1,NN                                                       B3150...
      RU=DABS(UVEC(I)-UITER(I))                                            B3160...
      IF(RU-RUM) 7300,7260,7260                                            B3170...
 7260 RUM=RU                                                               B3180...
      IUWORS=I                                                             B3190...
 7300 CONTINUE                                                             B3200...
      IF(RUM.GT.RUMAX) IGOI=IGOI+2                                         B3210...
 7350 CONTINUE                                                             B3220...
      IF(IGOI.GT.0.AND.ITER.EQ.ITRMAX) ISTOP=-1                            B3230...
      IF(IGOI.GT.0.AND.ISTOP.EQ.0) GOTO 1100                               B3240...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - B3250...
C.....END ITERATION - - - - - - - - - - - - - - - - - - - - - - - - - - - - B3260...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - B3270...
C                                                                          B3280...
 7500 CONTINUE                                                             B3290...
      IF(ISTOP.NE.-1.AND.IT.EQ.ITMAX) ISTOP=1                              B3300...
      IF(ISTOP.NE.-1.AND.TSEC.GE.TMAX) ISTOP=2                             B3310...
C                                                                          B3320...
C.....OUTPUT RESULTS FOR TIME STEP EACH NPRINT TIME STEPS                  B3330...
      IF(IT.GT.1.AND.MOD(IT,NPRINT).NE.0.AND.ISTOP.EQ.0) GOTO 8000         B3340...
C.....PRINT P AND/OR U, AND MAYBE SW AND/OR V                              B3350...
      CALL PRISOL(ML,ISTOP,IGOI,PVEC,UVEC,VMAG,VANG,SW)                    B3360...
C.....CALCULATE AND PRINT FLUID MASS AND/OR ENERGY OR SOLUTE MASS BUDGETB3370...
      IF(KBUDG.EQ.1)                                                       B3380...
     1   CALL BUDGET(ML,IBCT,VOL,SW,DSWDP,RHO,SOP,QIN,PVEC,PM1,           B3390...
     2   PBC,QPLITR,IPBC,IQSOP,POR,UVEC,UM1,UM2,UIN,QUIN,IQSOU,UBC,       B3400...
     3   CS1,CS2,CS3,SL,SR)                                               B3410...
C.....PLOT P RESULTS                                                       B3420...
      IF(KPLOTP.NE.1.OR.ML.EQ.2) GOTO 7680                                 B3430...
      CALL PLOT(1,2,X,Y,CC,INDEX,XX,YY,PVEC)                               B3440...
C.....PLOT U RESULTS                                                       B3450...
 7680 IF(KPLOTU.NE.1.OR.ML.EQ.1) GOTO 8000                                 B3460...
      NP=3                                                                 B3470...
      IF(ME.EQ.+1) NP=4                                                    B3480...
      CALL PLOT(1,NP,X,Y,CC,INDEX,XX,YY,UVEC)                              B3490...
 8000 CONTINUE                                                             B3500...
C                                                                          B3510...
C.....MAKE OBSERVATIONS AT OBSERVATION NODES EACH NOBCYC TIME STEPS        B3520...
      IF(NOBSN-1.GT.0) CALL OBSERV(1,IOBS,ITOBS,POBS,UOBS,OBSTIM,          B3530...
     1                             PVEC,UVEC,ISTOP)                      B3540...
C                                                                          B3550...
C.....STORE RESULTS FOR POSSIBLE RESTART OF SIMULATION EACH TIME STEP      B3560...
      IF(ISTORE.NE.1) GOTO 8150                                            B3570...
      CALL STORE(PVEC,UVEC,PM1,UM1,CS1,RCIT,SW,PBC)                        B3580...
C                                                                          B3590...
 8150 IF(ISTOP.EQ.0) GOTO 1000                                             B3600...
```

```
C ***************************************************************************B3610...
C.....END TIME STEP *******************************************************B3620...
C ***************************************************************************B3630...
C                                                                            B3640...
C                                                                            B3650...
C.....COMPLETE OUTPUT AND TERMINATE SIMULATION                               B3660...
      IF(ISTORE.EQ.1) WRITE(6,8100)                                          B3670...
 8100 FORMAT(//////11X,'*** LAST SOLUTION HAS BEEN STORED ',                 B3680...
     1   'ON UNIT 66 ***')                                                   B3690...
C                                                                            B3700...
C.....OUTPUT RESULTS OF OBSERVATIONS                                         B3710...
 8200 IF(NOBSN-1.GT.0) CALL OBSERV(2,IOBS,ITOBS,POBS,UOBS,OBSTIM,            B3720...
     1                             PVEC,UVEC,ISTOP)                         B3730...
C                                                                            B3740...
C.....OUTPUT END OF SIMULATION MESSAGE AND RETURN TO MAIN FOR STOP           B3750...
      IF(ISTOP.GT.0) GOTO 8400                                               B3760...
      IF(IGOI-2) 8230,8260,8290                                              B3770...
 8230 WRITE(6,8235)                                                          B3780...
 8235 FORMAT(/////////11X,'SIMULATION TERMINATED DUE TO ',                   B3790...
     1   'NON-CONVERGENT PRESSURE',                                          B3800...
     2              /11X,'********** ********** *** ** ',                   B3810...
     3   '************** ********')                                          B3820...
      RETURN                                                                 B3830...
 8260 IF(ME) 8262,8262,8266                                                  B3840...
 8262 WRITE(6,8264)                                                          B3850...
 8264 FORMAT(/////////11X,'SIMULATION TERMINATED DUE TO ',                   B3860...
     1   'NON-CONVERGENT CONCENTRATION',                                     B3870...
     2              /11X,'********** ********** *** ** ',                   B3880...
     3   '************** ************')                                      B3890...
      RETURN                                                                 B3900...
 8266 WRITE(6,8268)                                                          B3910...
 8268 FORMAT(/////////11X,'SIMULATION TERMINATED DUE TO ',                   B3920...
     1   'NON-CONVERGENT TEMPERATURE',                                       B3930...
     2              /11X,'********** ********** *** ** ',                   B3940...
     3   '************** ***********')                                       B3950...
      RETURN                                                                 B3960...
 8290 IF(ME) 8292,8292,8296                                                  B3970...
 8292 WRITE(6,8294)                                                          B3980...
 8294 FORMAT(/////////11X,'SIMULATION TERMINATED DUE TO ',                   B3990...
     1   'NON-CONVERGENT PRESSURE AND CONCENTRATION',                        B4000...
     2              /11X,'********** ********** *** ** ',                   B4010...
     3   '************** ******** *** ************')                        B4020...
      RETURN                                                                 B4030...
 8296 WRITE(6,8298)                                                          B4040...
 8298 FORMAT(/////////11X,'SIMULATION TERMINATED DUE TO ',                   B4050...
     1   'NON-CONVERGENT PRESSURE AND TEMPERATURE',                          B4060...
     2              /11X,'********** ********** *** ** ',                   B4070...
     3   '************** ******** *** ***********')                         B4080...
      RETURN                                                                 B4090...
C                                                                            B4100...
 8400 IF(ISTOP.EQ.2) GOTO 8500                                               B4110...
      WRITE(6,8450)                                                          B4120...
 8450 FORMAT(/////////11X,'SUTRA SIMULATION TERMINATED AT COMPLETION ',      B4130...
     1   'OF TIME STEPS'/                                                    B4140...
     2              11X,'***** ********** ********** ** ********** ',       B4150...
     3   '** **** *****')                                                    B4160...
      RETURN                                                                 B4170...
 8500 WRITE(6,8550)                                                          B4180...
 8550 FORMAT(/////////11X,'SUTRA SIMULATION TERMINATED AT COMPLETION ',      B4190...
     1   'OF TIME PERIOD'/                                                   B4200...
```

```
C     SUBROUTINE        S  U  T  R  A           SUTRA - VERSION 1284-2D B10.....


     2              11X,'***** ********** ********** ** ********** ',  B4210...
     3   '** **** ******')                                              B4220...
      RETURN                                                           B4230...
C                                                                      B4240...
      END                                                              B4250...
```

```
C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....


C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....
C                                                                       C20.....
C *** PURPOSE :                                                         C30.....
C ***  TO INPUT ,OUTPUT, AND ORGANIZE A MAJOR PORTION OF                C40.....
C ***  UNIT-5 INPUT DATA (DATASET 5 THROUGH DATASET 15B)                C50.....
C                                                                       C60.....
      SUBROUTINE INDAT1(X,Y,THICK,POR,ALMAX,ALMIN,ATAVG,PERMXX,PERMXY,  C70.....
     1   PERMYX,PERMYY,PANGLE,SOP)                                      C80.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               C90.....
      CHARACTER*10 ADSMOD                                               C100....
      CHARACTER*14 UTYPE(2)                                             C110....
      CHARACTER*6 STYPE(2)                                              C120....
      COMMON/MODSOR/ ADSMOD                                             C130....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            C140....
     1   NSOP,NSOU,NBCN                                                 C150....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        C160....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        C170....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        C180....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  C190....
      COMMON/ITERAT/ RPM,RPMAX,RUM,RUMAX,ITER,ITRMAX,IPWORS,IUWORS      C200....
      COMMON/TENSOR/ GRAVX,GRAVY                                        C210....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,C220....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2       C230....
      COMMON/SATPAR/ PCENT,SWRES,PCRES,SSLOPE,SINCPT                    C240....
      COMMON/KPRINT/ KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG      C250....
      DIMENSION X(NN),Y(NN),THICK(NN),POR(NN),SOP(NN)                   C260....
      DIMENSION PERMXX(NE),PERMXY(NE),PERMYX(NE),PERMYY(NE),PANGLE(NE), C270....
     1   ALMAX(NE),ALMIN(NE),ATAVG(NE)                                  C280....
      DATA UTYPE(1)/' TEMPERATURES '/,UTYPE(2)/'CONCENTRATIONS'/        C290....
      DATA STYPE(1)/'ENERGY'/,STYPE(2)/'SOLUTE'/                        C300....
C                                                                       C310....
      INSTOP=0                                                          C320....
C                                                                       C330....
C.....INPUT DATASET 5: NUMERICAL CONTROL PARAMETERS                     C340....
      READ(5,50) UP,GNU                                                 C350....
   50 FORMAT(G10.0,G15.0)                                               C360....
      WRITE(6,70) UP,GNU                                                C370....
   70 FORMAT(////11X,'N U M E R I C A L   C O N T R O L   D A T A'//    C380....
     1   11X,F15.5,5X,'"UPSTREAM WEIGHTING" FACTOR'/                    C390....
     2   11X,1PD15.4,5X,'SPECIFIED PRESSURE BOUNDARY CONDITION FACTOR') C400....
C                                                                       C410....
C.....INPUT DATASET 6: TEMPORAL CONTROL AND SOLUTION CYCLING DATA       C420....
      READ(5,100) ITMAX,DELT,TMAX,ITCYC,DTMULT,DTMAX,NPCYC,NUCYC        C430....
  100 FORMAT(I5,2G15.0,I10,G10.0,G15.0,2I5)                             C440....
      WRITE(6,120) ITMAX,DELT,TMAX,ITCYC,DTMULT,DTMAX,NPCYC,NUCYC       C450....
  120 FORMAT(1H1////11X,'T E M P O R A L   C O N T R O L   A N D   ',   C460....
     1   'S C L U T I O N   C Y C L I N G   D A T A',                   C470....
     2   //11X,I15,5X,'MAXIMUM ALLOWED NUMBER OF TIME STEPS'           C480....
     3   /11X,1PD15.4,5X,'INITIAL TIME STEP (IN SECONDS)'             C490....
     4   /11X,1PD15.4,5X,'MAXIMUM ALLOWED SIMULATION TIME (IN SECONDS)' C500....
     5   //11X,I15,5X,'TIME STEP MULTIPLIER CYCLE (IN TIME STEPS)'     C510....
     6   /11X,0PF15.5,5X,'MULTIPLICATION FACTOR FOR TIME STEP CHANGE'  C520....
     7   /11X,1PD15.4,5X,'MAXIMUM ALLOWED TIME STEP (IN SECONDS)'      C530....
     8   //11X,I15,5X,'FLOW SOLUTION CYCLE (IN TIME STEPS)'           C540....
     9   /11X,I15,5X,'TRANSPORT SOLUTION CYCLE (IN TIME STEPS)')       C550....
      IF(NPCYC.GE.1.AND.NUCYC.GE.1) GOTO 140                            C560....
      WRITE(6,130)                                                      C570....
  130 FORMAT(//11X,'* * * * ERROR DETECTED :  BOTH NPCYC AND ',         C580....
     1   'NUCYC MUST BE SET GREATER THAN OR EQUAL TO 1.')               C590....
      INSTOP=INSTOP-1                                                   C600....
```

```
C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....


  140 IF(NPCYC.EQ.1.OR.NUCYC.EQ.1) GOTO 160                             C610....
      WRITE(6,150)                                                      C620....
  150 FORMAT(//11X,'* * * * ERROR DETECTED :  EITHER NPCYC OR ',        C630....
     1   'NUCYC MUST BE SET TO 1.')                                     C640....
      INSTOP=INSTOP-1                                                   C650....
  160 CONTINUE                                                          C660....
C.....SET MAXIMUM ALLOWED TIME STEPS IN SIMULATION FOR                  C670....
C        STEADY-STATE FLOW AND STEADY-STATE TRANSPORT SOLUTION MODES    C680....
      IF(ISSFLO.EQ.1) THEN                                              C690....
       NPCYC=ITMAX+1                                                    C700....
       NUCYC=1                                                          C710....
       ENDIF                                                            C720....
      IF(ISSTRA.EQ.1) ITMAX=1                                           C730....
C                                                                       C740....
C.....INPUT DATASET 7: OUTPUT CONTROLS AND OPTIONS                      C750....
      READ(5,170) NPRINT,KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG  C760....
  170 FORMAT(16I5)                                                      C770....
      WRITE(6,172) NPRINT                                               C780....
  172 FORMAT(////11X,'O U T P U T   C O N T R O L S   A N D            C790....
     1   'O P T I O N S'//11X,I6,5X,'PRINTED OUTPUT CYCLE ',            C800....
     2   '(IN TIME STEPS)')                                             C810....
      IF(KNODAL.EQ.+1) WRITE(6,174)                                     C820....
      IF(KNODAL.EQ.0) WRITE(6,175)                                      C830....
  174 FORMAT(/11X,'- PRINT NODE COORDINATES, THICKNESSES AND',          C840....
     1   ' POROSITIES')                                                 C850....
  175 FORMAT(/11X,'- CANCEL PRINT OF NODE COORDINATES, THICKNESSES AND',C860....
     1   ' POROSITIES')                                                 C870....
      IF(KELMNT.EQ.+1) WRITE(6,176)                                     C880....
      IF(KELMNT.EQ.0) WRITE(6,177)                                      C890....
  176 FORMAT(11X,'- PRINT ELEMENT PERMEABILITIES AND DISPERSIVITIES')   C900....
  177 FORMAT(11X,'- CANCEL PRINT OF ELEMENT PERMEABILITIES AND ',       C910....
     1   'DISPERSIVITIES')                                              C920....
      IF(KINCID.EQ.+1) WRITE(6,178)                                     C930....
      IF(KINCID.EQ.0) WRITE(6,179)                                      C940....
  178 FORMAT(11X,'- PRINT NODE AND PINCH NODE INCIDENCES IN EACH ',     C950....
     1   'ELEMENT')                                                     C960....
  179 FORMAT(11X,'- CANCEL PRINT OF NODE AND PINCH NODE INCIDENCES ',   C970....
     1   'IN EACH ELEMENT')                                             C980....
      IF(KPLOTP.EQ.+1) WRITE(6,180)                                     C990....
      IF(KPLOTP.EQ.0) WRITE(6,181)                                      C1000...
  180 FORMAT(/11X,'- PLOT PRESSURES ON EACH TIME STEP WITH OUTPUT')     C1010...
  181 FORMAT(/11X,'- CANCEL PLOT OF PRESSURES')                         C1020...
      IME=2                                                             C1030...
      IF(ME.EQ.+1) IME=1                                                C1040...
      IF(KPLOTU.EQ.+1) WRITE(6,182) UTYPE(IME)                          C1050...
      IF(KPLOTU.EQ.0) WRITE(6,183) UTYPE(IME)                           C1060...
  182 FORMAT(11X,'- PLOT ',A14,' ON EACH TIME STEP WITH OUTPUT')        C1070...
  183 FORMAT(11X,'- CANCEL PLOT OF ',A14)                               C1080...
      IF(KVEL.EQ.+1) WRITE(6,184)                                       C1090...
      IF(KVEL.EQ.0) WRITE(6,185)                                        C1100...
  184 FORMAT(/11X,'- CALCULATE AND PRINT VELOCITIES AT ELEMENT ',       C1110...
     1   'CENTROIDS ON EACH TIME STEP WITH OUTPUT')                     C1120...
  185 FORMAT(/11X,'- CANCEL PRINT OF VELOCITIES')                       C1130...
      IF(KBUDG.EQ.+1) WRITE(6,186) STYPE(IME)                           C1140...
      IF(KBUDG.EQ.0) WRITE(6,187)                                       C1150...
  186 FORMAT(/11X,'- CALCULATE AND PRINT FLUID AND ',A6,' BUDGETS ',    C1160...
     1   'ON EACH TIME STEP WITH OUTPUT')                               C1170...
  187 FORMAT(/11X,'- CANCEL PRINT OF BUDGETS')                          C1180...
C                                                                       C1190...
C.....INPUT DATASET 8: ITERATION CONTROLS                               C1200...
```

```
C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....


      READ(5,190) ITRMAX,RPMAX,RUMAX                                    C1210...
  190 FORMAT(I10,2G10.0)                                                C1220...
      IF(ITRMAX-1) 192,192,194                                          C1230...
  192 WRITE(6,193)                                                      C1240...
  193 FORMAT(////11X,'I T E R A T I O N   C O N T R O L   D A T A',     C1250...
     1   //11X,'  NON-ITERATIVE SOLUTION')                             C1260...
      GOTO 196                                                          C1270...
  194 WRITE(6,195) ITRMAX,RPMAX,RUMAX                                   C1280...
  195 FORMAT(////11X,'I T E R A T I O N   C O N T R O L   D A T A',     C1290...
     1   //11X,I15,5X,'MAXIMUM NUMBER OF ITERATIONS PER TIME STEP',    C1300...
     2   /11X,1PD15.4,5X,'ABSOLUTE CONVERGENCE CRITERION FOR FLOW',     C1310...
     3   ' SOLUTION'/11X,1PD15.4,5X,'ABSOLUTE CONVERGENCE CRITERION',   C1320...
     4   ' FOR TRANSPORT SOLUTION')                                     C1330...
  196 CONTINUE                                                          C1340...
C                                                                       C1350...
C.....INPUT DATASET 9: FLUID PROPERTIES                                 C1360...
      READ(5,200) COMPFL,CW,SIGMAW,RHOWO,URHOWO,DRWDU,VISCO             C1370...
C.....INPUT DATASET 10: SOLID MATRIX PROPERTIES                         C1380...
      READ(5,200) COMPMA,CS,SIGMAS,RHOS                                 C1390...
  200 FORMAT(8G10.0)                                                    C1400...
      IF(ME.EQ.+1)                                                      C1410...
     1   WRITE(6,210) COMPFL,COMPMA,CW,CS,VISCO,RHOS,RHOWO,DRWDU,URHOWO,C1420...
     2                SIGMAW,SIGMAS                                     C1430...
  210 FORMAT(1H1////11X,'C O N S T A N T   P R O P E R T I E S   O F',  C1440...
     1   '   F L U I D   A N D   S O L I D   M A T R I X'               C1450...
     2   //11X,1PD15.4,5X,'COMPRESSIBILITY OF FLUID'/11X,1PD15.4,5X,    C1460...
     3   'COMPRESSIBILITY OF POROUS MATRIX'//11X,1PD15.4,5X,            C1470...
     4   'SPECIFIC HEAT CAPACITY OF FLUID',/11X,1PD15.4,5X,             C1480...
     5   'SPECIFIC HEAT CAPACITY OF SOLID GRAIN'//13X,'FLUID VISCOSITY',C1490...
     6   ' IS CALCULATED BY SUTRA AS A FUNCTION OF TEMPERATURE IN ',    C1500...
     7   'UNITS OF [kg/(m*s)]'//11X,1PD15.4,5X,'VISCO, CONVERSION ',    C1510...
     8   'FACTOR FOR VISCOSITY UNITS,  [desired units] = VISCO*',       C1520...
     9   '[kg/(m*s)]'//11X,1PD15.4,5X,'DENSITY OF A SOLID GRAIN'        C1530...
     *   //13X,'FLUID DENSITY, RHOW'/13X,'CALCULATED BY ',             C1540...
     1   'SUTRA IN TERMS OF TEMPERATURE, U, AS:'/13X,'RHOW = RHOWO + ', C1550...
     2   'DRWDU*(U-URHOWO)'//11X,1PD15.4,5X,'FLUID BASE DENSITY, RHOWO' C1560...
     3   /11X,1PD15.4,5X,'COEFFICIENT OF DENSITY CHANGE WITH ',         C1570...
     4   'TEMPERATURE, DRWDU'/11X,1PD15.4,5X,'TEMPERATURE, URHOWO, ',   C1580...
     5   'AT WHICH FLUID DENSITY IS AT BASE VALUE, RHOWO'               C1590...
     6   //11X,1PD15.4,5X,'THERMAL CONDUCTIVITY OF FLUID'               C1600...
     7   /11X,1PD15.4,5X,'THERMAL CONDUCTIVITY OF SOLID GRAIN')         C1610...
      IF(ME.EQ.-1)                                                      C1620...
     1   WRITE(6,220) COMPFL,COMPMA,VISCO,RHOS,RHOWO,DRWDU,URHOWO,SIGMAWC1630...
  220 FORMAT(1H1////11X,'C O N S T A N T   P R O P E R T I E S   O F',  C1640...
     1   '   F L U I D   A N D   S O L I D   M A T R I X'               C1650...
     2   //11X,1PD15.4,5X,'COMPRESSIBILITY OF FLUID'/11X,1PD15.4,5X,    C1660...
     3   'COMPRESSIBILITY OF POROUS MATRIX'                             C1670...
     4   //11X,1PD15.4,5X,'FLUID VISCOSITY'                            C1680...
     4   //11X,1PD15.4,5X,'DENSITY OF A SOLID GRAIN'                   C1690...
     5   //13X,'FLUID DENSITY, RHOW'/13X,'CALCULATED BY ',             C1700...
     6   'SUTRA IN TERMS OF SOLUTE CONCENTRATION, U, AS:',              C1710...
     7   /13X,'RHOW = RHOWO + DRWDU*(U-URHOWO)'                         C1720...
     8   //11X,1PD15.4,5X,'FLUID BASE DENSITY, RHOWO'                  C1730...
     9   /11X,1PD15.4,5X,'COEFFICIENT OF DENSITY CHANGE WITH ',         C1740...
     *   'SOLUTE CONCENTRATION, DRWDU'                                  C1750...
     1   /11X,1PD15.4,5X,'SOLUTE CONCENTRATION, URHOWO, ',              C1760...
     4   'AT WHICH FLUID DENSITY IS AT BASE VALUE, RHOWO'               C1770...
     5   //11X,1PD15.4,5X,'MOLECULAR DIFFUSIVITY OF SOLUTE IN FLUID')  C1780...
C                                                                       C1790...
C.....INPUT DATASET 11: ADSORPTION PARAMETERS                           C1800...
```

```
C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....


      READ(5,230) ADSMOD,CHI1,CHI2                                      C1810...
  230 FORMAT(A10,2G10.0)                                                C1820...
      IF(ME.EQ.+1) GOTO 248                                             C1830...
      IF(ADSMOD.EQ.'NONE      ') GOTO 234                               C1840...
      WRITE(6,232) ADSMOD                                               C1850...
  232 FORMAT(////11X,'A D S O R P T I O N   P A R A M E T E R S'        C1860...
     1   //16X,A10,5X,'EQUILIBRIUM SORPTION ISOTHERM')                  C1870...
      GOTO 236                                                          C1880...
  234 WRITE(6,235)                                                      C1890...
  235 FORMAT(////11X,'A D S O R P T I O N   P A R A M E T E R S'        C1900...
     1   //16X,'NON-SORBING SOLUTE')                                    C1910...
  236 IF((ADSMOD.EQ.'NONE ').OR.(ADSMOD.EQ.'LINEAR    ').OR.            C1920...
     1   (ADSMOD.EQ.'FREUNDLICH').OR.(ADSMOD.EQ.'LANGMUIR  ')) GOTO 238 C1930...
      WRITE(6,237)                                                      C1940...
  237 FORMAT(//11X,'* * * * ERROR DETECTED :  TYPE OF SORPTION MODEL ', C1950...
     1   'IS NOT SPECIFIED CORRECTLY.'/11X,'CHECK FOR TYPE AND ',       C1960...
     2   'SPELLING, AND THAT TYPE IS LEFT-JUSTIFIED IN INPUT FIELD')    C1970...
      INSTOP=INSTOP-1                                                   C1980...
  238 IF(ADSMOD.EQ.'LINEAR    ') WRITE(6,242) CHI1                      C1990...
  242 FORMAT(11X,1PD15.4,5X,'LINEAR DISTRIBUTION COEFFICIENT')          C2000...
      IF(ADSMOD.EQ.'FREUNDLICH') WRITE(6,244) CHI1,CHI2                 C2010...
  244 FORMAT(11X,1PD15.4,5X,'FREUNDLICH DISTRIBUTION COEFFICIENT'       C2020...
     1   /11X,1PD15.4,5X,'SECOND FREUNDLICH COEFFICIENT')               C2030...
      IF(ADSMOD.EQ.'FREUNDLICH'.AND.CHI2.LE.0.0D0) THEN                 C2040...
       WRITE(6,245)                                                     C2050...
  245  FORMAT(11X,'* * * * ERROR DETECTED :  SECOND COEFFICIENT ',      C2060...
     1    'MUST BE GREATER THAN ZERO')                                  C2070...
       INSTOP=INSTOP-1                                                  C2080...
       ENDIF                                                            C2090...
      IF(ADSMOD.EQ.'LANGMUIR  ') WRITE(6,246) CHI1,CHI2                 C2100...
  246 FORMAT(11X,1PD15.4,5X,'LANGMUIR DISTRIBUTION COEFFICIENT'         C2110...
     1   /11X,1PD15.4,5X,'SECOND LANGMUIR COEFFICIENT')                 C2120...
C                                                                       C2130...
C.....INPUT DATASET 12: PRODUCTION OF ENERGY OR SOLUTE MASS             C2140...
  248 READ(5,200) PRODF0,PRODS0,PRODF1,PRODS1                           C2150...
      IF(ME.EQ.-1) WRITE(6,250) PRODF0,PRODS0,PRODF1,PRODS1             C2160...
  250 FORMAT(////11X,'P R O D U C T I O N   A N D   D E C A Y   O F   ',C2170...
     1   'S P E C I E S   M A S S'//13X,'PRODUCTION RATE (+)'/13X,      C2180...
     2   'DECAY RATE (-)'//11X,1PD15.4,5X,'ZERO-ORDER RATE OF SOLUTE ', C2190...
     3   'MASS PRODUCTION/DECAY IN FLUID'/11X,1PD15.4,5X,               C2200...
     4   'ZERO-ORDER RATE OF ADSORBATE MASS PRODUCTION/DECAY IN ',      C2210...
     5   'IMMOBILE PHASE'/11X,1PD15.4,5X,'FIRST-ORDER RATE OF SOLUTE ', C2220...
     3   'MASS PRODUCTION/DECAY IN FLUID'/11X,1PD15.4,5X,               C2230...
     4   'FIRST-ORDER RATE OF ADSORBATE MASS PRODUCTION/DECAY IN ',     C2240...
     5   'IMMOBILE PHASE')                                              C2250...
      IF(ME.EQ.+1) WRITE(6,260) PRODF0,PRODS0                           C2260...
  260 FORMAT(////11X,'P R O D U C T I O N   A N D   L O S S   O F   ',  C2270...
     1   'E N E R G Y'//13X,'PRODUCTION RATE (+)'/13X,                  C2280...
     2   'LOSS RATE (-)'//11X,1PD15.4,5X,'ZERO-ORDER RATE OF ENERGY ',  C2290...
     3   'PRODUCTION/LOSS IN FLUID'/11X,1PD15.4,5X,                     C2300...
     4   'ZERO-ORDER RATE OF ENERGY PRODUCTION/LOSS IN ',               C2310...
     5   'SOLID GRAINS')                                                C2320...
C.....SET PARAMETER SWITCHES FOR EITHER ENERGY OR SOLUTE TRANSPORT      C2330...
      IF(ME) 272,272,274                                                C2340...
C     FOR SOLUTE TRANSPORT:                                             C2350...
  272 CS=0.0D0                                                          C2360...
      CW=1.0D0                                                          C2370...
      SIGMAS=0.0D0                                                      C2380...
      GOTO 278                                                          C2390...
C     FOR ENERGY TRANSPORT:                                             C2400...
```

```
  274 ADSMOD='NONE                                                          C2410...
      CHI1=0.000                                                            C2420...
      CHI2=0.000                                                            C2430...
      PRODF1=0.000                                                          C2440...
      PRODS1=0.000                                                          C2450...
C     DIVIDE SIGMA TO CANCEL MULTIPLICATION BY RHOW*CW                      C2460...
C        IN SUBROUTINE ELEMEN.                                              C2470...
      RCO=RHOWO*CW                                                          C2480...
      SIGMAW=SIGMAW/RCO                                                     C2490...
      SIGMAS=SIGMAS/RCO                                                     C2500...
  278 CONTINUE                                                              C2510...
C                                                                           C2520...
C.....INPUT DATASET 13: ORIENTATION OF COORDINATES TO GRAVITY               C2530...
      READ(5,200) GRAVX,GRAVY                                               C2540...
      WRITE(6,320) GRAVX,GRAVY                                              C2550...
  320 FORMAT(////11X,'C O O R D I N A T E   O R I E N T A T I O N   ',      C2560...
     1   'T O   G R A V I T Y'//13X,'COMPONENT OF GRAVITY VECTOR',          C2570...
     2   /13X,'IN +X DIRECTION, GRAVX'/11X,1PD15.4,5X,                      C2580...
     3   'GRAVX = -GRAV * D(ELEVATION)/DX'//13X,'COMPONENT OF GRAVITY',     C2590...
     4   ' VECTOR'/13X,'IN +Y DIRECTION, GRAVY'/11X,1PD15.4,5X,             C2600...
     5   'GRAVY = -GRAV * D(ELEVATION)/DY')                                 C2610...
C                                                                           C2620...
C.....INPUT DATASETS 14A AND 14B: NODEWISE DATA                             C2630...
      READ(5,330) SCALX,SCALY,SCALTH,PORFAC                                 C2640...
  330 FORMAT(5X,4G10.0)                                                     C2650...
      DO 450 I=1,NN                                                         C2660...
      READ(5,400) II,X(II),Y(II),THICK(II),POR(II)                          C2670...
  400 FORMAT(I5,4G10.0)                                                     C2680...
      X(II)=X(II)*SCALX                                                     C2690...
      Y(II)=Y(II)*SCALY                                                     C2700...
      THICK(II)=THICK(II)*SCALTH                                            C2710...
      POR(II)=POR(II)*PORFAC                                                C2720...
C     SET SPECIFIC PRESSURE STORATIVITY, SOP.                               C2730...
  450 SOP(II)=(1.DO-POR(II))*COMPMA+POR(II)*COMPFL                          C2740...
  460 IF(KNODAL.EQ.0) WRITE(6,469) SCALX,SCALY,SCALTH,PORFAC                C2750...
  469 FORMAT(1H1////11X,'N O D E   I N F O R M A T I O N'//16X,             C2760...
     1   'PRINTOUT OF NODE COORDINATES, THICKNESSES AND POROSITIES ',       C2770...
     2   'CANCELLED.'//16X,'SCALE FACTORS :'/33X,1PD15.4,5X,'X-SCALE'/      C2780...
     1   33X,1PD15.4,5X,'Y-SCALE'/33X,1PD15.4,5X,'THICKNESS FACTOR'/        C2790...
     2   33X,1PD15.4,5X,'POROSITY FACTOR')                                  C2800...
      IF(KNODAL.EQ.+1) WRITE(6,470) (I,X(I),Y(I),THICK(I),POR(I),I=1,NN)    C2810...
  470 FORMAT(1H1//11X,'N O D E   I N F O R M A T I O N'//13X,              C2820...
     1   'NODE',7X,'X',16X,'Y',17X,'THICKNESS',6X,'POROSITY'//              C2830...
     2   (11X,I6,3(3X,1PD14.5),6X,0PF8.5))                                  C2840...
C                                                                           C2850...
C.....INPUT DATASETS 15A AND 15B: ELEMENTWISE DATA                          C2860...
      READ(5,490) PMAXFA,PMINFA,ANGFAC,ALMAXF,ALMINF,ATAVGF                 C2870...
  490 FORMAT(10X,6G10.0)                                                    C2880...
      IF(KELMNT.EQ.+1) WRITE(6,500)                                         C2890...
  500 FORMAT(1H1//11X,'E L E M E N T   I N F O R M A T I O N'//             C2900...
     1   11X,'ELEMENT',4X,'MAXIMUM',9X,'MINIMUM',12X,                       C2910...
     2   'ANGLE BETWEEN',3X,'     MAXIMUM',5X,'   MINIMUM',5X,              C2920...
     3   '   AVERAGE'/22X,'PERMEABILITY',4X,'PERMEABILITY',4X,              C2930...
     4   '+X-DIRECTION AND',3X,'LONGITUDINAL',3X,'LONGITUDINAL'3X,          C2940...
     5   ' TRANSVERSE'/50X,'MAXIMUM PERMEABILITY',3X,'DISPERSIVITY',        C2950...
     6   3X,'DISPERSIVITY',3X,'DISPERSIVITY'/58X,'(IN DEGREES)'//)          C2960...
      DO 550 LL=1,NE                                                        C2970...
      READ(5,510) L,PMAX,PMIN,ANGLEX,ALMAX(L),ALMIN(L),ATAVG(L)             C2980...
  510 FORMAT(I10,6G10.0)                                                    C2990...
      PMAX=PMAX*PMAXFA                                                      C3000...
```

```
C     SUBROUTINE        I  N  D  A  T  1        SUTRA - VERSION 1284-2D C10.....


      PMIN=PMIN*PMINFA                                                  C3010...
      ANGLEX=ANGLEX*ANGFAC                                              C3020...
      ALMAX(L)=ALMAX(L)*ALMAXF                                          C3030...
      ALMIN(L)=ALMIN(L)*ALMINF                                          C3040...
      ATAVG(L)=ATAVG(L)*ATAVGF                                          C3050...
      IF(KELMNT.EQ.+1) WRITE(6,520) L,PMAX,PMIN,ANGLEX,                 C3060...
     1   ALMAX(L),ALMIN(L),ATAVG(L)                                     C3070...
  520 FORMAT(11X,I7,2X,2(1PD14.5,2X),8X,4(0PF10.3,5X))                  C3080...
C                                                                       C3090...
C.....ROTATE PERMEABILITY FROM MAXIMUM/MINIMUM TO X/Y DIRECTIONS        C3100...
      RADIAX=1.745329D-2*ANGLEX                                         C3110...
      SINA=DSIN(RADIAX)                                                 C3120...
      COSA=DCOS(RADIAX)                                                 C3130...
      SINA2=SINA*SINA                                                   C3140...
      COSA2=COSA*COSA                                                   C3150...
      PERMXX(L)=PMAX*COSA2+PMIN*SINA2                                   C3160...
      PERMYY(L)=PMAX*SINA2+PMIN*COSA2                                   C3170...
      PERMXY(L)=(PMAX-PMIN)*SINA*COSA                                   C3180...
      PERMYX(L)=PERMXY(L)                                               C3190...
      PANGLE(L)=RADIAX                                                  C3200...
  550 CONTINUE                                                          C3210...
      IF(KELMNT.EQ.0)                                                   C3220...
     1   WRITE(6,569) PMAXFA,PMINFA,ANGFAC,ALMAXF,ALMINF,ATAVGF         C3230...
  569 FORMAT(////11X,'E L E M E N T   I N F O R M A T I O N'//          C3240...
     1   16X,'PRINTOUT OF ELEMENT PERMEABILITIES AND DISPERSIVITIES ',  C3250...
     2   'CANCELLED.'//16X,'SCALE FACTORS :'/33X,1PD15.4,5X,'MAXIMUM ', C3260...
     1   'PERMEABILITY FACTOR'/33X,1PD15.4,5X,'MINIMUM PERMEABILITY ',  C3270...
     2   'FACTOR'/33X,1PD15.4,5X,'ANGLE FROM +X TO MAXIMUM DIRECTION',  C3280...
     3   ' FACTOR'/33X,1PD15.4,5X,'MAXIMUM LONGITUDINAL DISPERSIVITY',  C3290...
     4   ' FACTOR'/33X,1PD15.4,5X,'MINIMUM LONGITUDINAL DISPERSIVITY',  C3300...
     5   ' FACTOR'/33X,1PD15.4,5X,'TRANSVERSE DISPERSIVITY FACTOR')     C3310...
C                                                                       C3320...
C.....END SIMULATION FOR CORRECTIONS TO UNIT-5 DATA IF NECESSARY        C3330...
      IF(INSTOP.EQ.0) GOTO 1000                                         C3340...
      WRITE(6,999)                                                      C3350...
  999 FORMAT(////////11X,'PLEASE CORRECT INPUT DATA AND RERUN.',        C3360...
     1   ///22X,'S I M U L A T I O N   H A L T E D',                    C3370...
     2     /22X,'**********************   ************')               C3380...
      ENDFILE(6)                                                        C3390...
      STOP                                                              C3400...
C                                                                       C3410...
C                                                                       C3420...
 1000 RETURN                                                            C3430...
      END                                                               C3440...
```

```
C     SUBROUTINE        P  L  O  T              SUTRA - VERSION 1284-2D D10.....

C     SUBROUTINE        P  L  O  T              SUTRA - VERSION 1284-2D D10.....
C                                                                       D20.....
C *** PURPOSE :                                                         D30.....
C ***  TO READ PLOT SET-UP DATA, AND TO PLOT THE FINITE ELEMENT         D40.....
C ***  MESH, THE PRESSURE SOLUTION AND/OR THE CONCENTRATION OR          D50.....
C ***  TEMPERATURE SOLUTION ON THE PRINTED OUTPUT PAGE.                 D60.....
C                                                                       D70.....
      SUBROUTINE PLOT (ICALL,NP,X,Y,CC,INDEX,XX,YY,CVEC)                D80.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               D90.....
      COMMON/KPRINT/ KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG      D100....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        D110....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  D120....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            D130....
     1   NSOP,NSOU,NBCN                                                 D140....
      CHARACTER*1 PRNT(122),SYM(17),BLANK(60)                           D150....
      DOUBLE PRECISION NX(500),NY(14)                                   D160....
      CHARACTER*4 DIGIT(82),VF1(6),VF2(6),VF3(7)                        D170....
      DIMENSION K(10),N(10)                                             D180....
      CHARACTER*30 TITLE(1,4)                                           D190....
      DIMENSION X(NN),Y(NN),CC(NN),XX(NN),YY(NN),INDEX(NN),CVEC(NN)     D200....
      DATA SYM/'1','2','3','4','5','6','7','8','9','0',' ','.','Y','*', D210....
     1'E','-','+'/,PRNT/122*' '/,BLANK/60*' '/,NDS/1/                   D220....
      DATA DIGIT/'1','2','3','4','5','6','7','8','9','10','11','12','13'D230....
     1,'14','15','16','17','18','19','20','21','22','23','24','25','26' D240....
     2,'27','28','29','30','31','32','33','34','35','36','37','38','39',D250....
     2'40',      '41','82','83','84','85','86','87','88','89','90','91',D260....
     1'92','93','94','95','96','97','98','99','100','101','102','103', 'D270....
     2104','105','106','107','108','109','110','111','112','113','114','D280....
     3115','116','117','118','119','120','121','122'/                  D290....
      DATA TITLE/' * * * *  N O D E S  * * * *  ',                      D300....
     1   ' * * *  PRESSURE/PBASE  * * * ',                            D310....
     2   ' * *  CONCENTRATION/CBASE  * *',                             D320....
     3   '  * *  TEMPERATURE/TBASE  * * '/                             D330....
      DATA VF1/'(1H ',',',',',    ',',A1,F',',10.2',',')'/              D340....
      DATA VF2/'(1H ',',',',',    ',',A1,1',',X,A8',',')'/              D350....
      DATA VF3/'(1HO',',',',',    ',',A1,F',',3.0,',',12F1',',0.2)'/    D360....
C                                                                       D370....
C                                                                       D380....
      IF(ICALL) 1100,1100,1                                             D390....
C.....READ PLOT SETUP DATA (DATASET 16)                                 D400....
 1100 READ(5,1200) IDIREC,NLINPI,NCHAPI,NCHAPL                          D410....
 1200 FORMAT(4I5)                                                       D420....
      PLTWID=(DBLE(NCHAPL)-13.0D0)/DBLE(NCHAPI)                         D430....
      N1=NLINPI                                                         D440....
      N2=NCHAPI                                                         D450....
      N3=NCHAPL                                                         D460....
      XN1=1.D0/(2.D0*N1)                                                D470....
      NXS=1                                                             D480....
      NYS=1                                                             D490....
      NINY=PLTWID                                                       D500....
      K(1)=NN                                                           D510....
C                                                                       D520....
      IF(KPLOTP.NE.1) GOTO 1400                                         D530....
      READ(5,1300) PBASE                                                D540....
 1300 FORMAT(D13.0)                                                     D550....
 1400 IF(KPLOTU.NE.1) GOTO 1500                                         D560....
      READ(5,1300) UBASE                                                D570....
 1500 CONTINUE                                                          D580....
      WRITE(6,1520) IDIREC,NLINPI,NCHAPI,NCHAPL                         D590....
 1520 FORMAT(////11X,'P L O T   I N F O R M A T I O N'//                 D600....
```

```
C     SUBROUTINE        P  L  O  T                SUTRA - VERSION 1284-2D D10.....


     1   13X,'PLOT ORIENTATION'/                                          D610....
     2   I15,5X,'IDIREC....=-1 SMALL PLOT ACROSS PAPER, =+1 ',            D620....
     3   'LARGE PLOT ALONG PAPER'//13X,'LINE PRINTER CHARACTERISTICS'/    D630....
     4   11X,I15,5X,'NUMBER OF OUTPUT ',                                  D640....
     5   'LINES PER INCH'/11X,I15,5X,'NUMBER OF OUTPUT CHARACTERS',       D650....
     6   ' PER INCH'/11X,I15,5X,'MAXIMUM NUMBER OF OUTPUT ',              D660....
     7   'CHARACTERS PER LINE')                                           D670....
      IF(KPLOTP.NE.1) GOTO 1540                                           D680....
      WRITE(6,1530) PBASE                                                 D690....
 1530 FORMAT(//13X,'PRESSURE PLOT DATA'/11X,1PD15.4,5X,                   D700....
     1   'PBASE....PLOTTED PRESSURE VALUE IS PRESSURE/PBASE')             D710....
 1540 IF(KPLOTU.NE.1) GOTO 1580                                           D720....
      IF(ME) 1550,1550,1560                                               D730....
 1550 WRITE(6,1555) UBASE                                                 D740....
 1555 FORMAT(//13X,'CONCENTRATION PLOT DATA'/11X,1PD15.4,5X,              D750....
     1   'UBASE....PLOTTED CONCENTRATION VALUE IS CONCENTRATION/UBASE')   D760....
      GOTO 1580                                                           D770....
 1560 WRITE(6,1565) UBASE                                                 D780....
 1565 FORMAT(//13X,'TEMPERATURE PLOT DATA'/11X,1PD15.4,5X,                D790....
     1   'UBASE....PLOTTED TEMPERATURE VALUE IS TEMPERATURE/UBASE')       D800....
 1580 WRITE(6,1590)                                                       D810....
 1590 FORMAT(//31X,'THE THREE DIGITS PLOTTED ARE THE ONE TO THE LEFT,',   D820....
     1   /31X,'AND THE TWO TO THE RIGHT OF THE DECIMAL POINT')            D830....
C                                                                         D840....
C.....SET LONGER PLOT AXIS DOWN (IDIREC=+1)                               D850....
C                  OR ACROSS PAPER (IDIREC=-1)                            D860....
      SMALLX=0.0D0                                                        D870....
      SMALLY=0.0D0                                                        D880....
      BIGX=0.0D0                                                          D890....
      BIGY=0.0D0                                                          D900....
      DO 1600 I=1,NN                                                      D910....
      IF(X(I).GT.BIGX) BIGX=X(I)                                          D920....
      IF(X(I).LT.SMALLX) SMALLX=X(I)                                      D930....
      IF(Y(I).GT.BIGY) BIGY=Y(I)                                          D940....
 1600 IF(Y(I).LT.SMALLY) SMALLY=Y(I)                                      D950....
      XRANGE=BIGX-SMALLX                                                  D960....
      YRANGE=BIGY-SMALLY                                                  D970....
      TENTHX=XRANGE/10.0D0                                                D980....
      TENTHY=YRANGE/10.0D0                                                D990....
      IF(XRANGE.GE.YRANGE.AND.IDIREC.NE.-1) KKKKK=+1                      D1000...
      IF(XRANGE.GE.YRANGE.AND.IDIREC.EQ.-1) KKKKK=-1                      D1010...
      IF(XRANGE.LT.YRANGE.AND.IDIREC.NE.-1) KKKKK=-1                      D1020...
      IF(XRANGE.LT.YRANGE.AND.IDIREC.EQ.-1) KKKKK=+1                      D1030...
      IF(KKKKK.EQ.-1) GOTO 344                                            D1040...
      XMIN=SMALLX-TENTHX                                                  D1050...
      XMAX=BIGX+TENTHX                                                    D1060...
      YMIN=SMALLY-TENTHY                                                  D1070...
C     YMAX=BIGY+TENTHY                                                    D1080...
      GOTO 345                                                            D1090...
  344 XMIN=SMALLY-TENTHY                                                  D1100...
      XMAX=BIGY+TENTHY                                                    D1110...
      YMIN=SMALLX-TENTHX                                                  D1120...
C     YMAX=BIGX+TENTHX                                                    D1130...
  345 CONTINUE                                                            D1140...
      XRANGE=XRANGE*1.2D0                                                 D1150...
      YRANGE=YRANGE*1.2D0                                                 D1160...
      IF(KKKKK.EQ.+1) NINX=(NINY/YRANGE)*XRANGE+0.5D0                     D1170...
      IF(KKKKK.EQ.-1) NINX=(NINY/XRANGE)*YRANGE+0.5D0                     D1180...
C                                                                         D1190...
C     INITIALIZE PLOT COORDINATES...ROTATE IF REQUIRED (WHEN KKKKK=-1)    D1200...
```

```
C     SUBROUTINE        P  L  O  T              SUTRA - VERSION 1284-2D D10.....


C     (NOTE: YY PLOTS ACROSS PAGE, XX PLOTS ALONG PAGE)                 D1210...
      IF (KKKKK.EQ.-1) GOTO 361                                         D1220...
      DO 362 I=1,NN                                                     D1230...
      XX(I)= X(I)                                                       D1240...
      YY(I)= Y(I)                                                       D1250...
  362 INDEX(I)= I                                                       D1260...
      GOTO 368                                                          D1270...
  361 DO 363 I=1,NN                                                     D1280...
      XX(I)=+Y(I)                                                       D1290...
      YY(I)=+X(I)                                                       D1300...
C     NOTE THAT THE SIGN OF YY IS REVERSED LATER                        D1310...
C     IN ORDER TO COMPLETE THE ROTATION                                 D1320...
  363 INDEX(I)= I                                                       D1330...
  368 CONTINUE                                                          D1340...
C                                                                       D1350...
C.....INITIALIZE VARIABLES                                             D1360...
      NXD=NXS*NINX                                                      D1370...
      NYD=NYS*NINY                                                      D1380...
      IF(NXD.GE.((NYD+1)/2)) GOTO 11                                    D1390...
      NINX=1+((NYD-1)/(2*NXS))                                          D1400...
      NXD=NXS*NINX                                                      D1410...
   11 XSF=XRANGE/NXD                                                    D1420...
      YSF=YRANGE/NYD                                                    D1430...
      IF(KKKKK.EQ.+1) GOTO 12                                           D1440...
      XSF=YRANGE/NXD                                                    D1450...
      YSF=XRANGE/NYD                                                    D1460...
   12 CONTINUE                                                          D1470...
      N4=NXD*N1+1                                                       D1480...
      N5=NXD+1                                                          D1490...
      N6=NYD+1                                                          D1500...
      N7=N1*NINX                                                        D1510...
      N8=N2*NYD+1                                                       D1520...
      N9=N2*NINY                                                        D1530...
      NR=N8-1                                                           D1540...
      NA=N4/2-2                                                         D1550...
      NBB=N4/2+4                                                        D1560...
      NC=(N3-N8-10)/2                                                   D1570...
      ND=NC+N8                                                          D1580...
      NEE=MAXO(N5,N6)                                                   D1590...
      VF1(3)=DIGIT(ND-40)                                               D1600...
      VF2(3)=DIGIT(ND-40)                                               D1610...
      VF3(3)=DIGIT(NC)                                                  D1620...
C.....ARRANGE EACH DATA SET IN DESCENDING VALUES OF X                   D1630...
      DO 90 L=1,NDS                                                     D1640...
      NNN=K(L)                                                          D1650...
      DO 30 I=1,NNN                                                     D1660...
      BIG=XX(I)                                                         D1670...
      KK=I                                                              D1680...
      DO 20 J=I,NNN                                                     D1690...
      IF(XX(J).GT.BIG) GO TO 15                                         D1700...
      GO TO 20                                                          D1710...
   15 BIG=XX(J)                                                         D1720...
      KK=J                                                              D1730...
   20 CONTINUE                                                          D1740...
      TEMPI=YY(I)                                                       D1750...
      TEMPII=XX(I)                                                      D1760...
      TEMP3=INDEX(I)                                                    D1770...
      YY(I)=YY(KK)                                                      D1780...
      XX(I)=XX(KK)                                                      D1790...
      INDEX(I)=INDEX(KK)                                                D1800...
```

```
      INDEX(KK)=TEMP3                                                   D1810...
      YY(KK)=TEMPI                                                      D1820...
   30 XX(KK)=TEMPII                                                     D1830...
   90 CONTINUE                                                          D1840...
C                                                                       D1850...
C.....COMPUTE LABEL NUMBERS FOR X AND Y AXES                            D1860...
      DO 100 I=1,NEE                                                    D1870...
      NNX=N5-I                                                          D1880...
      NNY=N6-I                                                          D1890...
      IF(NNY.LT.0) GO TO 95                                             D1900...
      NY(I)=YSF*NNY+YMIN                                                D1910...
      IF(KKKKK.EQ.-1)   NY(I)=YMIN+(I-1)*YSF                            D1920...
   95 IF(NNX.LT.0) GO TO 100                                            D1930...
      NX(I)=XSF*NNX+XMIN                                                D1940...
  100 CONTINUE                                                          D1950...
C                                                                       D1960...
C.....SET UP PLOT OF MESH                                               D1970...
      DO 105 I=1,NN                                                     D1980...
  105 CVEC(I)=I*00.010000                                               D1990...
C                                                                       D2000...
C                                                                       D2010...
C                                                                       D2020...
C                                                                       D2030...
C.....ENTRY FOR PRESSURE AND CONCENTRATION OR TEMPERATURE PLOTS         D2040...
C.....--------                                                          D2050...
    1 CONTINUE                                                          D2060...
C.....--------                                                          D2070...
C .....NORMALIZE VARIABLE TO BE PLOTTED                                 D2080...
      CCNORM=1.000                                                      D2090...
      IF(NP.EQ.2) CCNORM=PBASE                                          D2100...
      IF(NP.GT.2) CCNORM=UBASE                                          D2110...
      DO 2 I=1,NN                                                       D2120...
    2 CC(I)=CVEC(INDEX(I))/CCNORM                                       D2130...
C                                                                       D2140...
C.....INITIALIZE VARIABLES                                              D2150...
      Z=XMAX                                                            D2160...
      WRITE (6,40)                                                      D2170...
      DO 10 I=1,NDS                                                     D2180...
   10 N(I)=1                                                            D2190...
      DO 210 I=1,N4                                                     D2200...
C                                                                       D2210...
C.....LOCATE X AXES                                                     D2220...
      IF (I.EQ.1.OR.I.EQ.N4) GO TO 110                                  D2230...
      DO 114 J=1,N8,N9                                                  D2240...
  114 PRNT(J)=SYM(15)                                                   D2250...
C                                                                       D2260...
C.....LOCATE Y AXES                                                     D2270...
      IF ((I-1)/N1*N1.NE.I-1) GO TO 117                                 D2280...
  115 PRNT(1)=SYM(14)                                                   D2290...
      PRNT( N8)=SYM(14)                                                 D2300...
  117 IF((I-1)/N7*N7.NE.I-1) GO TO 130                                  D2310...
      DO 118 J=2,NR                                                     D2320...
      IF((J-1)/N9*N9.EQ.J-1)PRNT(J)=SYM(17)                             D2330...
  118 IF((J-1)/N9*N9.NE.J-1)PRNT(J)=SYM(16)                             D2340...
      GO TO 130                                                         D2350...
  110 DO 120 J=1,N8                                                     D2360...
      IF ((J-1)/N2*N2.EQ.J-1) PRNT(J)=SYM(14)                           D2370...
  120 IF ((J-1)/N2*N2.NE.J-1) PRNT(J)=SYM(16)                           D2380...
C                                                                       D2390...
C.....COMPUTE LOCATION OF POINTS                                        D2400...
```

```
  130 DO 150 J=1,NDS                                                    D2410...
  135 IF (N(J).EQ.K(J)+1) GO TO 150                                     D2420...
      IF(I.GT.1) GO TO 137                                              D2430...
      IF(XX(N(J)).LE.Z+XN1*XSF) GO TO 137                               D2440...
      N(J)=N(J)+1                                                       D2450...
      GO TO 135                                                         D2460...
  137 IF (XX(N(J)).LE.Z+XN1*XSF.AND.XX(N(J)).GE.Z-XN1*XSF) GO TO 140    D2470...
      GO TO 150                                                         D2480...
C 140 M=NR+0.5D0- ((YY(N(J))-YMIN)*N2)/YSF                              D2490...
  140 DELYC=      ((YY(N(J))-YMIN)*N2)/YSF                              D2500...
      M=NR+0.5D0 - DELYC                                                D2510...
C                                                                       D2520...
C     REVERSE SIGN OF YY (I.E. REVERSE PLOTTING DIRECTION) IF           D2530...
C     GRAPH IS TO BE TRANSPOSED....                                     D2540...
      IF(KKKKK.EQ.-1)   M=0.5D0  + DELYC                                D2550...
C                                                                       D2560...
      IF(M.LT.0.OR.M.GT.NR) GO TO 145                                   D2570...
      IF(CC(N(J)))142,146,147                                           D2580...
  142 IF(M.NE.0) PRNT(M)=SYM(16)                                        D2590...
      NUM=(-CC(N(J))+.005D0)*10.D0                                      D2600...
      GO TO 141                                                         D2610...
  147 NUM=(CC(N(J))+0.005D0)*100.D0                                     D2620...
      IF (NUM.GT.999) NUM=MOD(NUM,1000)                                 D2630...
  141 IF(NUM.LT.100) GO TO 143                                          D2640...
      INDX3=NUM/100                                                     D2650...
      IF (M.NE.0.AND.CC(N(J)).GT.0.) PRNT(M)=SYM(INDX3)                 D2660...
      NUM=NUM-INDX3*100                                                 D2670...
  143 INDX1=MOD(NUM,10)                                                 D2680...
      IF(INDX1.EQ.0) INDX1=10                                           D2690...
      INDX2=NUM/10                                                      D2700...
      IF(INDX2.EQ.0) INDX2=10                                           D2710...
      GO TO 144                                                         D2720...
  146 INDX1=14                                                          D2730...
      INDX2=14                                                          D2740...
  144 PRNT(M+1)=SYM(INDX2)                                              D2750...
      PRNT(M+2)=SYM(INDX1)                                              D2760...
  145 N(J)=N(J)+1                                                       D2770...
      IF (N(J).EQ.K(J)+1) GO TO 150                                     D2780...
      IF (XX(N(J)).LE.Z+XN1*XSF.AND.XX(N(J)).GE.Z-XN1*XSF) GO TO 140    D2790...
  150 CONTINUE                                                          D2800...
C                                                                       D2810...
C.....PRINT AXES,LABELS, AND POINTS                                     D2820...
C     IF (I-NA.EQ.0) GO TO 170                                          D2830...
C     IF (I-NBB.EQ.0) GO TO 180                                         D2840...
      IF ((I-1)/N1*N1-(I-1)) 190,160,190                                D2850...
  160 WRITE (6,VF1)(BLANK(J),J=1,NC),(PRNT(J),J=1,N8),NX(1+(I-1)/N1)    D2860...
      GO TO 200                                                         D2870...
C 170 WRITE (6,VF2)(BLANK(J),J=1,NC),(PRNT(J),J=1,N8)                   D2880...
C     GO TO 200                                                         D2890...
C 180 WRITE (6,VF2)(BLANK(J),J=1,NC),(PRNT(J),J=1,N8)                   D2900...
C     GO TO 200                                                         D2910...
  190 WRITE (6,VF2)(BLANK(J),J=1,NC),(PRNT(J),J=1,N8)                   D2920...
C                                                                       D2930...
C.....COMPUTE NEW VALUE FOR Z AND INITIALIZE PRNT                       D2940...
  200 Z=Z-2.D0*XN1*XSF                                                  D2950...
      DO 210 J=1,N8                                                     D2960...
  210 PRNT(J)=SYM(11)                                                   D2970...
C                                                                       D2980...
C.....NUMBER AND LABEL Y AXIS AND PRINT TITLE                           D2990...
      WRITE (6,VF3)(BLANK(J),J=1,NC),(NY(I),I=1,N6)                     D3000...
```

```
C     SUBROUTINE        P  L  O  T              SUTRA - VERSION 1284-2D D10....

      WRITE (6,80)  (TITLE(1,NP))                                       D3010...
C                                                                       D3020...
      RETURN                                                            D3030...
C                                                                       D3040...
C.....FORMATS                                                           D3050...
   40 FORMAT ('1')                                                      D3060...
   80 FORMAT ('0',41X,1A30)                                             D3070...
      END                                                               D3080...
```

```
C     SUBROUTINE        S  O  U  R  C  E        SUTRA - VERSION 1284-2D E10.....


C     SUBROUTINE        S  O  U  R  C  E        SUTRA - VERSION 1284-2D E10.....
C                                                                       E20.....
C *** PURPOSE :                                                         E30.....
C ***  TO READ AND ORGANIZE FLUID MASS SOURCE DATA AND ENERGY OR        E40.....
C ***  SOLUTE MASS SOURCE DATA.                                         E50.....
C                                                                       E60.....
      SUBROUTINE SOURCE(QIN,UIN,IQSOP,QUIN,IQSOU,IQSOPT,IQSOUT)        E70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                              E80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,           E90.....
     1   NSOP,NSOU,NBCN                                               E100....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,       E110....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                E120....
      DIMENSION QIN(NN),UIN(NN),IQSOP(NSOP),QUIN(NN),IQSOU(NSOU)       E130....
C                                                                      E140....
C.....NSOPI IS ACTUAL NUMBER OF FLUID SOURCE NODES                     E150....
C.....NSOUI IS ACTUAL NUMBER OF SOLUTE MASS OR ENERGY SOURCE NODES     E160....
      NSOPI=NSOP-1                                                     E170....
      NSOUI=NSOU-1                                                     E180....
      IQSOPT=1                                                         E190....
      IQSOUT=1                                                         E200....
      NIQP=0                                                           E210....
      NIQU=0                                                           E220....
      IF(NSOPI.EQ.0) GOTO 1000                                         E230....
      IF(ME) 50,50,150                                                 E240....
   50 WRITE(6,100)                                                     E250....
  100 FORMAT(1H1////11X,'F L U I D   S O U R C E   D A T A'            E260....
     1   ////11X,'**** NODES AT WHICH FLUID INFLOWS OR OUTFLOWS ARE ',  E270....
     2   'SPECIFIED ****'//11X,'NODE NUMBER',10X,                      E280....
     3   'FLUID INFLOW(+)/OUTFLOW(-)',5X,'SOLUTE CONCENTRATION OF'     E290....
     4   /11X,'(MINUS INDICATES',5X,'(FLUID MASS/SECOND)',             E300....
     5   12X,'INFLOWING FLUID'/12X,'TIME-VARYING',39X,                 E310....
     6   '(MASS SOLUTE/MASS WATER)'/12X,'FLOW RATE OR'/12X,            E320....
     7   'CONCENTRATION)'//)                                           E330....
      GOTO 300                                                         E340....
  150 WRITE(6,200)                                                     E350....
  200 FORMAT(1H1////11X,'F L U I D   S O U R C E   D A T A'            E360....
     1   ////11X,'**** NODES AT WHICH FLUID INFLOWS OR OUTFLOWS ARE ',  E370....
     2   'SPECIFIED ****'//11X,'NODE NUMBER',10X,                      E380....
     3   'FLUID INFLOW(+)/OUTFLOW(-)',5X,'TEMPERATURE [DEGREES CELCIUS]'E390....
     4   /11X,'(MINUS INDICATES',5X,'(FLUID MASS/SECOND)',12X,         E400....
     5   'OF INFLOWING FLUID'/12X,'TIME-VARYING'/12X,'FLOW OR'/12X,    E410....
     6   'TEMPERATURE)'//)                                             E420....
C                                                                      E430....
C.....INPUT DATASET 17                                                 E440....
  300 CONTINUE                                                         E450....
      READ(5,400) IQCP,QINC,UINC                                       E460....
  400 FORMAT(I10,2G15.0)                                               E470....
      IF(IQCP.EQ.0) GOTO 700                                           E480....
      NIQP=NIQP+1                                                      E490....
      IQSOP(NIQP)=IQCP                                                 E500....
      IF(IQCP.LT.0) IQSOPT=-1                                          E510....
      IQP=IABS(IQCP)                                                   E520....
      QIN(IQP)=QINC                                                    E530....
      UIN(IQP)=UINC                                                    E540....
      IF(IQCP.GT.0) GOTO 450                                           E550....
      WRITE(6,500) IQCP                                                E560....
      GOTO 600                                                         E570....
  450 IF(QINC.GT.0) GOTO 460                                           E580....
      WRITE(6,500) IQCP,QINC                                           E590....
      GOTO 600                                                         E600....
```

```
  460 WRITE(6,500) IQCP,QINC,UINC                                          E610....
  500 FORMAT(11X,I10,13X,1PE14.7,16X,1PE14.7)                              E620....
  600 GOTO 300                                                             E630....
  700 IF(NIQP.EQ.NSOPI) GOTO 890                                           E640....
C.....END SIMULATION IF THERE NEED BE CORRECTIONS TO DATASET 17            E650....
      WRITE(6,750) NIQP,NSOPI                                              E660....
  750 FORMAT(////11X,'THE NUMBER OF FLUID SOURCE NODES READ, ',I5,         E670....
     1   ' IS NOT EQUAL TO THE NUMBER SPECIFIED, ',I5////                  E680....
     2   11X,'PLEASE CORRECT DATA AND RERUN'////////                       E690....
     3   22X,'S I M U L A T I O N   H A L T E D'/                         E700....
     4   22X,'---------------------------------')                         E710....
      ENDFILE(6)                                                           E720....
      STOP                                                                 E730....
  890 IF(IQSOPT.EQ.-1) WRITE(6,900)                                        E740....
  900 FORMAT(////11X,'THE SPECIFIED TIME VARIATIONS ARE ',                 E750....
     1   'USER-PROGRAMMED IN SUBROUTINE  B C T I M E .')                   E760....
C                                                                          E770....
C                                                                          E780....
 1000 IF(NSOUI.EQ.0) GOTO 9000                                             E790....
      IF(ME) 1050,1050,1150                                                E800....
 1050 WRITE(6,1100)                                                        E810....
 1100 FORMAT(////////11X,'S O L U T E   S O U R C E   D A T A'             E820....
     1   ////11X,'**** NODES AT WHICH SOURCES OR SINKS OF SOLUTE ',        E830....
     2   'MASS ARE SPECIFIED ****'//11X,'NODE NUMBER',10X,                 E840....
     3   'SOLUTE SOURCE(+)/SINK(-)'/11X,'(MINUS INDICATES',5X,             E850....
     4   '(SOLUTE MASS/SECOND)'/12X,'TIME-VARYING'/12X,                    E860....
     5   'SOURCE OR SINK)'//)                                              E870....
      GOTO 1300                                                            E880....
 1150 WRITE(6,1200)                                                        E890....
 1200 FORMAT(////////11X,'E N E R G Y   S O U R C E   D A T A'             E900....
     1   ////11X,'**** NODES AT WHICH SOURCES OR SINKS OF ',              E910....
     2   'ENERGY ARE SPECIFIED ****'//11X,'NODE NUMBER',10X,               E920....
     3   'ENERGY SOURCE(+)/SINK(-)'/11X,'(MINUS INDICATES',5X,             E930....
     4   '(ENERGY/SECOND)'/12X,'TIME-VARYING'/12X,                         E940....
     5   'SOURCE OR SINK)'//)                                              E950....
C                                                                          E960....
C.....INPUT DATASET 18                                                     E970....
 1300 CONTINUE                                                             E980....
      READ(5,400) IQCU,QUINC                                               E990....
      IF(IQCU.EQ.0) GOTO 1700                                              E1000...
      NIQU=NIQU+1                                                          E1010...
      IQSOU(NIQU)=IQCU                                                     E1020...
      IF(IQCU.LT.0) IQSOUT=-1                                              E1030...
      IQU=IABS(IQCU)                                                       E1040...
      QUIN(IQU)=QUINC                                                      E1050...
      IF(IQCU.GT.0) GOTO 1450                                              E1060...
      WRITE(6,1500) IQCU                                                   E1070...
      GOTO 1600                                                            E1080...
 1450 WRITE(6,1500) IQCU,QUINC                                             E1090...
 1500 FORMAT(11X,I10,13X,1PE14.7)                                          E1100...
 1600 GOTO 1300                                                            E1110...
 1700 IF(NIQU.EQ.NSOUI) GOTO 1890                                          E1120...
C.....END SIMULATION IF THERE NEED BE CORRECTIONS TO DATASET 18            E1130...
      IF(ME) 1740,1740,1760                                                E1140...
 1740 WRITE(6,1750) NIQU,NSOUI                                             E1150...
 1750 FORMAT(////11X,'THE NUMBER OF SOLUTE SOURCE NODES READ, ',I5,        E1160...
     1      IS NOT EQUAL TO THE NUMBER SPECIFIED, ',I5////                 E1170...
     2   11X,'PLEASE CORRECT DATA AND RERUN'////////                       E1180...
     3   22X,'S I M U L A T I O N   H A L T E D'/                         E1190...
     4   22X,'---------------------------------')                         E1200...
```

```
      ENDFILE(6)                                                        E1210...
      STOP                                                              E1220...
 1760 WRITE(6,1770) NIQJ,NSOUI                                          E1230...
 1770 FORMAT(////11X,'THE NUMBER OF ENERGY SOURCE NODES READ, ',I5,     E1240...
     1   ' IS NOT EQUAL TO THE NUMBER SPECIFIED, ',I5////               E1250...
     2   11X,'PLEASE CORRECT DATA AND RERUN'////////                    E1260...
     3   22X,'S I M U L A T I O N   H A L T E D'/                      E1270...
     4   22X,'-----------------------------------')                    E1280...
      ENDFILE(6)                                                        E1290...
      STOP                                                              E1300...
 1890 IF(IQSOUT.EQ.-1) WRITE(6,900)                                     E1310...
C                                                                       E1320...
 9000 RETURN                                                            E1330...
C                                                                       E1340...
      END                                                               E1350...
```

```
C     SUBROUTINE        B  O  U  N  D           SUTRA - VERSION 1284-2D F10.....


C     SUBROUTINE        B  O  U  N  D           SUTRA - VERSION 1284-2D F10.....
C                                                                       F20.....
C *** PURPOSE :                                                         F30.....
C ***  TO READ AND ORGANIZE SPECIFIED PRESSURE DATA AND                 F40.....
C ***  SPECIFIED TEMPERATURE OR CONCENTRATION DATA.                     F50.....
C                                                                       F60.....
      SUBROUTINE BOUND(IPBC,PBC,IUBC,UBC,IPBCT,IUBCT)                   F70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               F80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            F90.....
     1   NSOP,NSOU,NBCN                                                 F100....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        F110....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  F120....
      DIMENSION IPBC(NBCN),PBC(NBCN),IUBC(NBCN),UBC(NBCN)               F130....
C                                                                       F140....
C                                                                       F150....
      IPBCT=1                                                           F160....
      IUBCT=1                                                           F170....
      ISTOPP=0                                                          F180....
      ISTOPU=0                                                          F190....
      IPU=0                                                             F200....
      WRITE(6,50)                                                       F210....
   50 FORMAT(1H1////11X,'B O U N D A R Y   C O N D I T I O N S')        F220....
      IF(NPBC.EQ.0) GOTO 400                                            F230....
      WRITE(6,100)                                                      F240....
  100 FORMAT(//11X,'**** NODES AT WHICH PRESSURES ARE',                 F250....
     1   ' SPECIFIED ****'/)                                            F260....
      IF(ME) 107,107,114                                                F270....
  107 WRITE(6,108)                                                      F280....
  108 FORMAT(11X,'     (AS WELL AS SOLUTE CONCENTRATION OF ANY'         F290....
     1   /16X,' FLUID INFLOW WHICH MAY OCCUR AT THE POINT'              F300....
     2   /16X,' OF SPECIFIED PRESSURE)'//12X,'NODE',18X,'PRESSURE',     F310....
     3   13X,'CONCENTRATION'//)                                         F320....
      GOTO 120                                                          F330....
  114 WRITE(6,115)                                                      F340....
  115 FORMAT(11X,'     (AS WELL AS TEMPERATURE [DEGREES CELCIUS] OF ANY'F350....
     1   /16X,' FLUID INFLOW WHICH MAY OCCUR AT THE POINT'              F360....
     2   /16X,' OF SPECIFIED PRESSURE)'//12X,'NODE',18X,                F370....
     2   'PRESSURE',13X,'  TEMPERATURE'//)                             F380....
C                                                                       F390....
C.....INPUT DATASET 14                                                  F400....
  120 IPU=IPU+1                                                         F410....
      READ(5,150) IPBC(IPU),PBC(IPU),UBC(IPU)                           F420....
  150 FORMAT(I5,2G20.0)                                                 F430....
      IF(IPBC(IPU).LT.0) IPBCT=-1                                       F440....
      IF(IPBC(IPU).EQ.0) GOTO 180                                       F450....
      IF(IPBC(IPU).GT.0) WRITE(6,160) IPBC(IPU),PBC(IPU),UBC(IPU)       F460....
      IF(IPBC(IPU).LT.0) WRITE(6,160) IPBC(IPU)                         F470....
  160 FORMAT(11X,I5,6X,1PD20.13,6X,1PD20.13)                            F480....
      GOTO 120                                                          F490....
  180 IPU=IPU-1                                                         F500....
      IP=IPU                                                            F510....
      IF(IP.EQ.NPBC) GOTO 200                                           F520....
      ISTOPP=1                                                          F530....
  200 IF(IPBCT.NE.-1) GOTO 400                                          F540....
      IF(ME) 205,205,215                                                F550....
  205 WRITE(6,206)                                                      F560....
  206 FORMAT(//12X,'TIME-DEPENDENT SPECIFIED PRESSURE'/12X,'OR INFLOW ',F570....
     1   'CONCENTRATION INDICATED'/12X,'BY NEGATIVE NODE NUMBER')       F580....
      GOTO 400                                                          F590....
  215 WRITE(6,216)                                                      F600....
```

```
  216 FORMAT(//11X,'TIME-DEPENDENT SPECIFIED PRESSURE'/12X,'OR INFLOW ',F610....
     1   'TEMPERATURE INDICATED'/12X,'BY NEGATIVE NODE NUMBER')          F620....
  400 IF(NUBC.EQ.0) GOTO 2000                                            F630....
C                                                                        F640....
      IF(ME) 500,500,550                                                 F650....
  500 WRITE(6,1000)                                                      F660....
 1000 FORMAT(////11X,'**** NODES AT WHICH SOLUTE CONCENTRATIONS ARE ',   F670....
     1   'SPECIFIED TO BE INDEPENDENT OF LOCAL FLOWS AND FLUID SOURCES',F680....
     2   ' ****'//12X,'NODE',13X,'CONCENTRATION'//)                     F690....
      GOTO 1120                                                          F700....
  550 WRITE(6,1001)                                                      F710....
 1001 FORMAT(////11X,'**** NODES AT WHICH TEMPERATURES ARE ',            F720....
     1   'SPECIFIED TO BE INDEPENDENT OF LOCAL FLOWS AND FLUID SOURCES',F730....
     2   ' ****'//12X,'NODE',15X,'TEMPERATURE'//)                       F740....
C                                                                        F750....
C.....INPUT DATASET 20                                                   F760....
 1120 IPU=IPU+1                                                          F770....
      READ(5,150) IUBC(IPU),UBC(IPU)                                     F780....
      IF(IUBC(IPU).LT.0) IUBCT=-1                                        F790....
      IF(IUBC(IPU).EQ.0) GOTO 1180                                       F800....
      IF(IUBC(IPU).GT.0) WRITE(6,1150) IUBC(IPU),UBC(IPU)                F810....
      IF(IUBC(IPU).LT.0) WRITE(6,1150) IUBC(IPU)                         F820....
 1150 FORMAT(11X,I5,6X,1PD20.13)                                         F830....
      GOTO 1120                                                          F840....
 1180 IPU=IPU-1                                                          F850....
      IU=IPU-IP                                                          F860....
      IF(IU.EQ.NUBC) GOTO 1200                                           F870....
      ISTOPU=1                                                           F880....
 1200 IF(IUBCT.NE.-1) GOTO 2000                                          F890....
      IF(ME) 1205,1205,1215                                              F900....
 1205 WRITE(6,1206)                                                      F910....
 1206 FORMAT(//12X,'TIME-DEPENDENT SPECIFIED CONCENTRATION'/12X,'IS ',   F920....
     1   'INDICATED BY NEGATIVE NODE NUMBER')                           F930....
      GOTO 2000                                                          F940....
 1215 WRITE(6,1216)                                                      F950....
 1216 FORMAT(//11X,'TIME-DEPENDENT SPECIFIED TEMPERATURE'/12X,'IS ',     F960....
     1   'INDICATED BY NEGATIVE NODE NUMBER')                           F970....
C                                                                        F980....
C.....END SIMULATION IF THERE NEED BE CORRECTIONS TO DATASET 19 OR 20    F990....
 2000 IF(ISTOPP.EQ.0.AND.ISTOPU.EQ.0) GOTO 6000                          F1000...
      IF(ISTOPP.EQ.1) WRITE(6,3000) IP,NPBC                              F1010...
 3000 FORMAT(////11X,'ACTUAL NUMBER OF SPECIFIED PRESSURE NODES',        F1020...
     1   ' READ, ',I5,', IS NOT EQUAL TO NUMBER SPECIFIED IN',          F1030...
     2   ' INPUT, ',I5)                                                 F1040...
      IF(ME) 3500,3500,4600                                              F1050...
 3500 IF(ISTOPU.EQ.1) WRITE(6,4000) IU,NUBC                              F1060...
 4000 FORMAT(////11X,'ACTUAL NUMBER OF SPECIFIED CONCENTRATION NODES',   F1070...
     1   ' READ, ',I5,', IS NOT EQUAL TO NUMBER SPECIFIED IN',          F1080...
     2   ' INPUT, ',I5)                                                 F1090...
      GOTO 4800                                                          F1100...
 4600 IF(ISTOPU.EQ.1) WRITE(6,4700) IU,NUBC                              F1110...
 4700 FORMAT(////11X,'ACTUAL NUMBER OF SPECIFIED TEMPERATURE NODES',     F1120...
     1   ' READ, ',I5,', IS NOT EQUAL TO NUMBER SPECIFIED IN',          F1130...
     2   ' INPUT, ',I5)                                                 F1140...
 4800 WRITE(6,5000)                                                      F1150...
 5000 FORMAT(////11X,'PLEASE CORRECT DATA AND RERUN.'////////            F1160...
     1   22X,'S I M U L A T I O N   H A L T E D'/                       F1170...
     2   22X,'-----------------------------------')                     F1180...
      ENDFILE(6)                                                         F1190...
      STOP                                                               F1200...
```

```
C     SUBROUTINE        B  O  U  N  D          SUTRA - VERSION 1284-2D F10.....


C                                                                       F1210...
 6000 IF(IPBCT.EQ.-1.OR.IUBCT.EQ.-1) WRITE(6,7000)                      F1220...
 7000 FORMAT(////11X,'THE SPECIFIED TIME VARIATIONS ARE ',              F1230...
     1   'USER-PROGRAMMED IN SUBROUTINE  B C T I M E .')                F1240...
C                                                                       F1250...
C                                                                       F1260...
      RETURN                                                            F1270...
      END                                                               F1280...
```

```
C     SUBROUTINE        O  B  S  E  R  V        SUTRA - VERSION 1284-2D G10.....
```


```
C     SUBROUTINE        O  B  S  E  R  V        SUTRA - VERSION 1284-2D G10.....
```


```
C                                                                       G20.....
C *** PURPOSE :                                                         G30.....
C ***  (1) TO READ AND ORGANIZE OBSERVATION NODE DATA                   G40.....
C ***  (2) TO MAKE OBSERVATIONS ON PARTICULAR TIME STEPS                G50.....
C ***  (3) TO OUTPUT OBSERVATIONS AFTER COMPLETION OF SIMULATION        G60.....
C                                                                       G70.....
      SUBROUTINE OBSERV(ICALL,IOBS,ITOBS,POBS,UOBS,OBSTIM,PVEC,UVEC,    G80.....
     1   ISTOP)                                                         G90.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               G100....
      CHARACTER*13 UNAME(2)                                             G110....
      CHARACTER*10 UNDERS                                               G120....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            G130....
     1   NSOP,NSOU,NBCN                                                 G140....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        G150....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  G160....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        G170....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        G180....
      COMMON/OBS/ NOBSN,NTOBSN,NOBCYC,ITCNT                             G190....
      DIMENSION INOB(16)                                                G200....
      DIMENSION IOBS(NOBSN),POBS(NOBSN,NTOBSN),UOBS(NOBSN,NTOBSN),      G210....
     1   OBSTIM(NTOBSN),ITOBS(NTOBSN),PVEC(NN),UVEC(NN)                 G220....
      DATA UNAME(1)/'CONCENTRATION'/,UNAME(2)/'  TEMPERATURE'/,          G230....
     1   UNDERS/'__________'/,                                          G240....
     1   ITCNT/0000/                                                    G250....
C                                                                       G260....
C.....NOBS IS ACTUAL NUMBER OF OBSERVATION NODES                        G270....
C.....NTOBS IS MAXIMUM NUMBER OF TIME STEPS WITH OBSERVATIONS           G280....
      NOBS=NOBSN-1                                                      G290....
      NTOBS=NTOBSN-2                                                    G300....
      IF(ICALL-1) 50,500,5000                                           G310....
C                                                                       G320....
C.....INITIALIZATION CALL                                               G330....
C.....INPUT DATASET 21                                                  G340....
   50 CONTINUE                                                          G350....
      JSTOP=0                                                           G360....
      WRITE(6,60)                                                       G370....
   60 FORMAT(////11X,'O B S E R V A T I O N   N O D E S')               G380....
      READ(5,65) NOBCYC                                                 G390....
   65 FORMAT(I10)                                                       G400....
      WRITE(6,70) NOBCYC                                                G410....
   70 FORMAT(//11X,'**** NODES AT WHICH OBSERVATIONS WILL BE MADE',     G420....
     1   ' EVERY',I5,' TIME STEPS ****'//)                              G430....
      NTOBSP=ITMAX/NOBCYC                                               G440....
      IF(NTOBSP.GT.NTOBS) WRITE(6,80) NTOBS,NTOBSP,ITMAX                G450....
   80 FORMAT(//11X,'-  W  A  R  N  I  N  G  -'/11X,                     G460....
     1   'NUMBER OF OBSERVATION STEPS SPECIFIED ',I5,                   G470....
     2   ', IS LESS THAN THE NUMBER POSSIBLE ',I5,','/                  G480....
     3   11X,'WITHIN THE MAXIMUM NUMBER OF ALLOWED TIME STEPS, ',I5,'.'/G490....
     4   11X,'PLEASE RECONFIRM THAT OBSERVATION COUNTS ARE CORRECT.'//) G500....
  100 READ(5,150) INOB                                                  G510....
  150 FORMAT(16I5)                                                      G520....
      IOB=0                                                             G530....
      DO 200 JJ=1,16                                                    G540....
      IF(INOB(JJ).EQ.0) GOTO 250                                        G550....
      IOB=IOB+1                                                         G560....
      IOBS(IOB)=INOB(JJ)                                                G570....
  200 CONTINUE                                                          G580....
      IF(IOB.LT.NOBS) GOTO 100                                          G590....
  250 IF(IOB.NE.NOBS) JSTOP=1                                           G600....
```

```
C     SUBROUTINE        O  B  S  E  R  V        SUTRA - VERSION 1284-2D G10.....


      WRITE(6,300) (IOBS(JJ),JJ=1,NOBS)                                 G610....
  300 FORMAT((11X,16(3X,I6)))                                           G620....
      IF(JSTOP.EQ.0) GOTO 400                                           G630....
C.....END SIMULATION IF CORRECTIONS ARE NECESSARY IN DATASET 21          G640....
      WRITE(6,350) IOB,NOBS                                             G650....
  350 FORMAT(////11X,'ACTUAL NUMBER OF OBSERVATION NODES',              G660....
     1   ' READ, ',I5,', IS NOT EQUAL TO NUMBER SPECIFIED IN',          G670....
     2   ' INPUT, ',I5////11X,'PLEASE CORRECT DATA AND RERUN.',         G680....
     3   /////////22X,'S I M U L A T I O N   H A L T E D'/              G690....
     4   22X,        '-----------------------------------')             G700....
      STOP                                                              G710....
  400 RETURN                                                            G720....
C                                                                       G730....
C.....MAKE OBSERVATIONS EACH NOBCYC TIME STEPS                          G740....
  500 CONTINUE                                                          G750....
      IF(MOD(IT,NOBCYC).NE.0.AND.IT.GT.1.AND.ISTOP.EQ.0) RETURN         G760....
      IF(IT.EQ.0) RETURN                                                G770....
      ITCNT=ITCNT+1                                                     G780....
      ITOBS(ITCNT)=IT                                                   G790....
      OBSTIM(ITCNT)=TSEC                                                G800....
      DO 1000 JJ=1,NOBS                                                 G810....
      I=IOBS(JJ)                                                        G820....
      POBS(JJ,ITCNT)=PVEC(I)                                            G830....
      UOBS(JJ,ITCNT)=UVEC(I)                                            G840....
 1000 CONTINUE                                                          G850....
      RETURN                                                            G860....
C                                                                       G870....
C.....OUTPUT OBSERVATIONS                                               G880....
 5000 CONTINUE                                                          G890....
      MN=2                                                              G900....
      IF(ME.EQ.-1) MN=1                                                 G910....
      JJ2=0                                                             G920....
      MLOOP=(NOBS+3)/4                                                  G930....
      DO 7000 LOOP=1,MLOOP                                              G940....
      JJ1=JJ2+1                                                         G950....
      JJ2=JJ2+4                                                         G960....
      IF(LOOP.EQ.MLOOP) JJ2=NOBS                                        G970....
      WRITE(6,5999) (IOBS(JJ),JJ=JJ1,JJ2)                               G980....
 5999 FORMAT(1H1///5X,'O  B  S  E  R  V  A  T  I  O  N     ',          G990....
     1   'N  O  D  E     D  A  T  A'///23X,4(:8X,'NODE ',I5,8X))        G1000...
      WRITE(6,6000) (UNDERS,JJ=JJ1,JJ2)                                 G1010...
 6000 FORMAT(                          23X,4(:8X,  A10   ,  8X))       G1020...
      WRITE(6,6001) (UNAME(MN),JJ=JJ1,JJ2)                              G1030...
 6001 FORMAT(/1X,'TIME STEP',4X,'TIME(SEC)',4(:2X,'PRESSURE',3X,A13))   G1040...
      DO 6500 ITT=1,ITCNT                                               G1050...
      WRITE(6,6100) ITOBS(ITT),OBSTIM(ITT),                             G1060...
     1   (POBS(JJ,ITT),JOBS(JJ,ITT),JJ=JJ1,JJ2)                         G1070...
 6100 FORMAT(5X,I5,1X,1PD12.5,8(1X,1PD12.5))                            G1080...
 6500 CONTINUE                                                          G1090...
 7000 CONTINUE                                                          G1100...
      RETURN                                                            G1110...
C                                                                       G1120...
C                                                                       G1130...
      END                                                               G1140...
```

```
C     SUBROUTINE        C  O  N  N  E  C        SUTRA - VERSION 1284-2D H10.....


C     SUBROUTINE        C  O  N  N  E  C        SUTRA - VERSION 1284-2D H10.....
C                                                                      H20.....
C *** PURPOSE :                                                        H30.....
C ***  TO READ ,ORGANIZE, AND CHECK DATA ON NODE INCIDENCES AND         H40.....
C ***  PINCH NODE INCIDENCES.                                          H50.....
C                                                                      H60.....
      SUBROUTINE CONNEC(IN,IPINCH)                                     H70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                              H80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,           H90.....
     1   NSOP,NSOU,NBCN                                                H100....
      COMMON/KPRINT/ KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG     H110....
      DIMENSION IN(NIN),IPINCH(NPINCH,3)                               H120....
      DIMENSION IIN(4),IEDGE(4),IK(8)                                  H130....
      DATA IK/1,2,2,3,3,4,4,1/                                         H140....
C                                                                      H150....
      ISTOP=0                                                          H160....
      IPIN=0                                                           H170....
      IF(KINCID.EQ.0) WRITE(6,1)                                       H180....
    1 FORMAT(1H1////11X,'M E S H   C O N N E C T I O N   D A T A'//     H190....
     1   16X,'PRINTOUT OF NODAL INCIDENCES AND PINCH NODE ',           H200....
     2   'CONNECTIONS CANCELLED.')                                     H210....
      IF(KINCID.EQ.+1) WRITE(6,2)                                      H220....
    2 FORMAT(1H1////11X,'M E S H   C O N N E C T I O N   D A T A',     H230....
     1   ///11X,'**** NODAL INCIDENCES ****'///)                       H240....
C                                                                      H250....
C.....INPUT DATASET 22 AND CHECK FOR ERRORS                            H260....
      DO 1000 L=1,NE                                                   H270....
      DO 4 I=1,4                                                       H280....
    4 IEDGE(I)=0                                                       H290....
      READ(5,10) LL,(IIN(II),II=1,4)                                   H300....
   10 FORMAT(5I6)                                                      H310....
C.....PREPARE NODE INCIDENCE LIST FOR MESH, IN.                        H320....
      DO 5 II=1,4                                                      H330....
      III=II+(L-1)*4                                                   H340....
    5 IN(III)=IIN(II)                                                  H350....
      IF(IABS(LL).EQ.L) GOTO 25                                        H360....
      WRITE(6,20) LL                                                   H370....
   20 FORMAT(11X,'ELEMENT ',I6,'INCIDENCE DATA IS NOT IN NUMERICAL',   H380....
     1   ' ORDER IN THE DATA SET')                                     H390....
      ISTOP=ISTOP+1                                                    H400....
   25 IF(LL.GE.0) GOTO 500                                             H410....
C                                                                      H420....
      READ(5,30) (IEDGE(I),I=1,4)                                      H430....
   30 FORMAT(4I6)                                                      H440....
C.....PREPARE PINCH NODE INCIDENCE LIST FOR MESH, IPINCH.              H450....
      DO 200 K=1,4                                                     H460....
      I=IEDGE(K)                                                       H470....
      IF(I) 200,200,100                                                H480....
  100 IPIN=IPIN+1                                                      H490....
      IPINCH(IPIN,1)=I                                                 H500....
      KK1=2*K-1                                                        H510....
      KK2=KK1+1                                                        H520....
      KKK1=IK(KK1)                                                     H530....
      KKK2=IK(KK2)                                                     H540....
      IPINCH(IPIN,2)=IIN(KKK1)                                         H550....
      IPINCH(IPIN,3)=IIN(KKK2)                                         H560....
  200 CONTINUE                                                         H570....
C                                                                      H580....
  500 M1=(L-1)*4+1                                                     H590....
      M4=M1+3                                                          H600....
```

```
C     SUBROUTINE        C  O  N  N  E  C        SUTRA - VERSION 1284-2D H10.....


      IF(KINCID.EQ.0) GOTO 1000                                         H610....
      WRITE(6,650) L,(IN(M),M=M1,M4)                                    H620....
  650 FORMAT(11X,'ELEMENT',I6,5X,' NODES AT : ',6X,'CORNERS ',          H630....
     1   5(1H*),4I6,1X,5(1H*))                                          H640....
      IF(LL.LT.0) WRITE(6,700)(IEDGE(M),M=1,4)                          H650....
  700 FORMAT(11X,'EDGES',4I6)                                           H660....
C                                                                       H670....
 1000 CONTINUE                                                          H680....
      IF(IPIN.EQ.0) GOTO 5000                                           H690....
      IF(IPIN.EQ.NPINCH-1) GOTO 1500                                    H700....
      WRITE(6,1450) IPIN,NPINCH                                         H710....
 1450 FORMAT(//////11X,'ACTUAL NUMBER OF PINCH NODES,',I4,              H720....
     1   ', DIFFERS FROM NUMBER ALLOWED AS SPECIFIED IN INPUT, ',I4//   H730....
     2   11X,'PLEASE CORRECT INPUT DATA AND/OR DIMENSIONS AND RERUN.'   H740....
     3   ////////22X,'S I M U L A T I O N   H A L T E D'/               H750....
     4            22X,'---------------------------------')              H760....
      STOP                                                              H770....
C                                                                       H780....
 1500 CONTINUE                                                          H790....
      IF(KINCID.EQ.0) GOTO 5000                                         H800....
      WRITE(6,3000)                                                     H810....
 3000 FORMAT(//////11X,'**** PINCH NODE CONNECTIONS ****'//7X,          H820....
     1   'PINCH NODE',17X,'CONNECTED NODES'///)                         H830....
      DO 4000 I=1,IPIN                                                  H840....
 4000 WRITE(6,4500) (IPINCH(I,NP),NP=1,3)                               H850....
 4500 FORMAT(11X,I6,10X,2I6)                                            H860....
C                                                                       H870....
C                                                                       H880....
 5000 RETURN                                                            H890....
      END                                                               H900....
```

```
C     SUBROUTINE        B  A  N  W  I  D        SUTRA - VERSION 1284-2D I10.....


C     SUBROUTINE        B  A  N  W  I  D        SUTRA - VERSION 1284-2D I10.....
C                                                                       I20.....
C *** PURPOSE :                                                         I30.....
C ***  TO CALCULATE AND CHECK BAND WIDTH OF FINITE ELEMENT MESH.         I40.....
C                                                                       I50.....
      SUBROUTINE BANWID(IN)                                             I60.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               I70.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            I80.....
     1   NSOP,NSOU,NBCN                                                 I90.....
      DIMENSION IN(NIN)                                                 I100....
C                                                                       I110....
      NBTEST=0                                                          I120....
      NDIF=0                                                            I130....
      II=0                                                              I140....
      WRITE(6,100)                                                      I150....
  100 FORMAT(////11X,'**** MESH ANALYSIS ****'//)                       I160....
C                                                                       I170....
C.....FIND ELEMENT WITH MAXIMUM DIFFERENCE IN NODE NUMBERS              I180....
      DO 2000 L=1,NE                                                    I190....
      II=II+1                                                           I200....
      IELO=IN(II)                                                       I210....
      IEHI=IN(II)                                                       I220....
      DO 1000 I=2,4                                                     I230....
      II=II+1                                                           I240....
      IF(IN(II).LT.IELO) IELO=IN(II)                                    I250....
 1000 IF(IN(II).GT.IEHI) IEHI=IN(II)                                    I260....
      NDIFF=IEHI-IELO                                                   I270....
      IF(NDIFF.GT.NDIF) THEN                                            I280....
       NDIF=NDIFF                                                       I290....
       LEM=L                                                            I300....
       ENDIF                                                            I310....
      NBL=2*NDIFF+1                                                     I320....
      IF(NBL.GT.NBI) WRITE(6,1500) L,NBL,NBI                            I330....
 1500 FORMAT(/13X,'ELEMENT ',I4,' HAS BANDWIDTH ',I5,                   I340....
     1   ' WHICH EXCEEDS INPUT BANDWIDTH ',I3)                          I350....
      IF(NBL.GT.NBI) NBTEST=NBTEST+1                                    I360....
 2000 CONTINUE                                                          I370....
C                                                                       I380....
C.....CALCULATE ACTUAL BAND WIDTH, NB.                                  I390....
      NB=2*NDIF+1                                                       I400....
      NBHALF=NDIF+1                                                     I410....
      WRITE(6,2500) NB,LEM,NBI                                          I420....
 2500 FORMAT(//13X,'ACTUAL MAXIMUM BANDWIDTH, ',I3,                     I430....
     1   ', WAS CALCULATED IN ELEMENT ',I4/13X,7(1H-),                  I440....
     2   'INPUT BANDWIDTH IS ',I3)                                      I450....
      IF(NBTEST.EQ.0) GOTO 3000                                         I460....
C                                                                       I470....
      WRITE(6,2800) NBTEST                                              I480....
 2800 FORMAT(//////13X,'INPUT BANDWIDTH IS EXCEEDED IN ',I4,' ELEMENTS',I490....
     1   /11X,'PLEASE CORRECT INPUT DATA AND RERUN.',                   I500....
     2   ////////22X,'S I M U L A T I O N   H A L T E D'/,              I510....
     3           22X,'-------------------------------')                 I520....
      ENDFILE(6)                                                        I530....
      STOP                                                              I540....
C                                                                       I550....
 3000 WRITE(6,4000)                                                     I560....
 4000 FORMAT(/////////1X,132(1H-)///42X,'E N D   O F   I N P U T   ',   I570....
     1   'F R O M   U N I T - 5'//132(1H-))                            I580....
      RETURN                                                            I590....
      END                                                               I600....
```

```
C     SUBROUTINE        N  C  H  E  C  K        SUTRA - VERSION 1284-2D J10.....
```


```
C     SUBROUTINE        N  C  H  E  C  K        SUTRA - VERSION 1284-2D J10.....
```


```
C                                                                       J20.....
C *** PURPOSE :                                                         J30.....
C ***  TO CHECK THAT PINCH NODES ARE NOT ASSIGNED SPECIFIED             J40.....
C ***  PRESSURES, CONCENTRATIONS, TEMPERATURES OR SOURCES.              J50.....
C                                                                       J60.....
      SUBROUTINE NCHECK(IPINCH,IQSOP,IQSOU,IPBC,IUBC)                   J70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               J80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            J90.....
     1   NSOP,NSOU,NBCN                                                 J100....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        J110....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  J120....
      DIMENSION JQPX(30),JQUX(30),JPX(30),JUX(30)                       J130....
      DIMENSION IPINCH(NPINCH,3),IQSOP(NSOP),IQSOU(NSOU),               J140....
     1   IPBC(NBCN),IUBC(NBCN)                                          J150....
C                                                                       J160....
      IQPX=0                                                            J170....
      IQUX=0                                                            J180....
      IPX=0                                                             J190....
      IUX=0                                                             J200....
      NPIN=NPINCH-1                                                     J210....
      NSOPI=NSOP-1                                                      J220....
      NSOUI=NSOU-1                                                      J230....
      DO 1000 I=1,NPIN                                                  J240....
      IPIN=IPINCH(I,1)                                                  J250....
C.....MATCH PINCH NODES WITH FLUID SOURCE NODES                         J260....
      IF(NSOPI.EQ.0) GOTO 200                                           J270....
      DO 100 IQP=1,NSOP                                                 J280....
      IF(IPIN-IABS(IQSOP(IQP))) 100,50,100                              J290....
   50 IQPX=IQPX+1                                                       J300....
      JQPX(IQPX)=IPIN                                                   J310....
  100 CONTINUE                                                          J320....
  200 IF(NSOUI.EQ.0) GOTO 400                                           J330....
C.....MATCH PINCH NODES WITH ENERGY OR SOLUTE MASS SOURCE NODES         J340....
      DO 300 IQU=1,NSOU                                                 J350....
      IF(IPIN-IABS(IQSOU(IQU))) 300,250,300                             J360....
  250 IQUX=IQUX+1                                                       J370....
      JQUX(IQUX)=IPIN                                                   J380....
  300 CONTINUE                                                          J390....
  400 IF(NPBC.EQ.0) GOTO 600                                            J400....
C.....MATCH PINCH NODES WITH SPECIFIED PRESSURE NODES                   J410....
      DO 500 IP=1,NPBC                                                  J420....
      IF(IPIN-IABS(IPBC(IP))) 500,450,500                               J430....
  450 IPX=IPX+1                                                         J440....
      JPX(IPX)=IPIN                                                     J450....
  500 CONTINUE                                                          J460....
  600 IF(NUBC.EQ.0) GOTO 1000                                           J470....
C.....MATCH PINCH NODES WITH SPECIFIED TEMPERATURE OR                   J480....
C        CONCENTRATION NODES                                            J490....
      DO 700 IU=1,NUBC                                                  J500....
      IUP=IU+NPBC                                                       J510....
      IF(IPIN-IABS(IUBC(IUP))) 700,650,700                              J520....
  650 IUX=IUX+1                                                         J530....
      JUX(IUX)=IPIN                                                     J540....
  700 CONTINUE                                                          J550....
 1000 CONTINUE                                                          J560....
C                                                                       J570....
C.....END SIMULATION IF CORRECTIONS TO UNIT-5 DATA ARE REQUIRED         J580....
      IF(IQPX.EQ.0) GOTO 1300                                           J590....
      WRITE(6,1250) (JQPX(I),I=1,IQPX)                                  J600....
```

```
 1250 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE SPECIFIED AS',     J610....
     1   ' FLUID SOURCE NODES : '/15X,2(20I6/))                         J620....
      WRITE(6,1251)                                                     J630....
 1251 FORMAT(/11X,'PLEASE REDISTRIBUTE SOURCES OR CHANGE THESE PINCH',  J640....
     1   ' NODES TO NORMAL CORNER MESH NODES AND THEN RERUN.')          J650....
 1300 IF(IQUX.EQ.0) GOTO 1400                                           J660....
      IF(ME.EQ.-1) WRITE(6,1350) (JQUX(I),I=1,IQUX)                     J670....
 1350 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE SPECIFIED AS',    J680....
     1   ' SOLUTE SOURCE NODES : '/15X,2(20I6/))                        J690....
      IF(ME.EQ.+1) WRITE(6,1355) (JQUX(I),I=1,IQUX)                     J700....
 1355 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE SPECIFIED AS',    J710....
     1   ' ENERGY SOURCE NODES : '/15X,2(20I6/))                        J720....
      WRITE(6,1251)                                                     J730....
 1400 IF(IPX.EQ.0) GOTO 1500                                            J740....
      WRITE(6,1450) (JPX(I),I=1,IPX)                                    J750....
 1450 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE INPUT AS',        J760....
     1   ' SPECIFIED PRESSURE NODES : ',/15X,2(20I6/))                  J770....
      WRITE(6,1451)                                                     J780....
 1451 FORMAT(/11X,'PLEASE REMOVE SPECIFIED PRESSURE RESTRICTION OR',    J790....
     1   ' CHANGE THESE PINCH NODES TO NORMAL CORNER MESH NODES AND',   J800....
     2   ' THEN RERUN.')                                                J810....
 1500 IF(ME) 1600,1600,1660                                             J820....
 1600 IF(IUX.EQ.0) GOTO 1680                                            J830....
      WRITE(6,1650) (JUX(I),I=1,IUX)                                    J840....
 1650 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE INPUT AS',        J850....
     1   ' SPECIFIED CONCENTRATION NODES : ',/15X,2(20I6/))             J860....
      WRITE(6,1651)                                                     J870....
 1651 FORMAT(/11X,'PLEASE REMOVE SPECIFIED CONCENTRATION RESTRICTION ', J880....
     1   'OR CHANGE THESE PINCH NODES TO NORMAL CORNER NODES AND',      J890....
     2   ' THEN RERUN.')                                                J900....
      GOTO 1680                                                         J910....
 1660 IF(IUX.EQ.0) GOTO 1680                                            J920....
      WRITE(6,1670) (JUX(I),I=1,IUX)                                    J930....
 1670 FORMAT(/////11X,'THE FOLLOWING NODES MAY NOT BE INPUT AS',        J940....
     1   ' SPECIFIED TEMPERATURE NODES : ',/15X,2(20I6/))               J950....
      WRITE(6,1671)                                                     J960....
 1671 FORMAT(/11X,'PLEASE REMOVE SPECIFIED TEMPERATURE RESTRICTION OR', J970....
     1   ' CHANGE THESE PINCH NODES TO NORMAL CORNER NODES AND',        J980....
     2   ' THEN RERUN.')                                                J990....
C                                                                       J1000...
 1680 IF(IQX+IPX+IUX) 1800,1800,1700                                    J1010...
 1700 WRITE(6,1750)                                                     J1020...
 1750 FORMAT(////////11X,'S I M U L A T I O N   H A L T E D'/,          J1030...
     1              11X,'-----------------------------------')          J1040...
      ENDFILE(6)                                                        J1050...
      STOP                                                              J1060...
C                                                                       J1070...
C                                                                       J1080...
 1800 RETURN                                                            J1090...
      END                                                               J1100...
```

```
C     SUBROUTINE        I  N  D  A  T  2        SUTRA - VERSION 1284-2D K10.....



C     SUBROUTINE        I  N  D  A  T  2        SUTRA - VERSION 1284-2D K10.....
C                                                                       K20.....
C *** PURPOSE :                                                         K30.....
C ***  TO READ INITIAL CONDITIONS FROM UNIT-55, AND TO                  K40.....
C ***  INITIALIZE DATA FOR EITHER WARM OR COLD START OF                 K50.....
C ***  THE SIMULATION.                                                  K60.....
C                                                                       K70.....
      SUBROUTINE INDAT2(PVEC,UVEC,PM1,UM1,UM2,CS1,CS2,CS3,SL,SR,RCIT,   K80.....
     1   SW,DSWDP,PBC,IPBC,IPBCT)                                       K90.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               K100....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            K110....
     1   NSOP,NSOU,NBCN                                                 K120....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        K130....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  K140....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        K150....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        K160....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,K170....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2       K180....
      DIMENSION PVEC(NN),UVEC(NN),PM1(NN),UM1(NN),UM2(NN),SL(NN),SR(NN),K190....
     1   CS1(NN),CS2(NN),CS3(NN),RCIT(NN),SW(NN),DSWDP(NN),             K200....
     2   PBC(NBCN),IPBC(NBCN)                                           K210....
C                                                                       K220....
C                                                                       K230....
      IF(IREAD) 500,500,620                                             K240....
C.....INPUT INITIAL CONDITIONS FOR WARM START (UNIT-55 DATA)            K250....
  500 READ(55,510) TSTART,DELTP,DELTU                                   K260....
  510 FORMAT(4G20.10)                                                   K270....
      READ(55,510) (PVEC(I),I=1,NN)                                     K280....
      READ(55,510) (UVEC(I),I=1,NN)                                     K290....
      READ(55,510) (PM1(I),I=1,NN)                                      K300....
      READ(55,510) (UM1(I),I=1,NN)                                      K310....
      READ(55,510) (CS1(I),I=1,NN)                                      K320....
      READ(55,510) (RCIT(I),I=1,NN)                                     K330....
      READ(55,510) (SW(I),I=1,NN)                                       K340....
      READ(55,510) (PBC(IPU),IPU=1,NBCN)                                K350....
C     CALL ZERO(CS2,NN,0.0D0)                                           K360....
C     CALL ZERO(CS3,NN,0.0D0)                                           K370....
      CALL ZERO(SL,NN,0.0D0)                                            K380....
      CALL ZERO(SR,NN,0.0D0)                                            K390....
      CALL ZERO(DSWDP,NN,0.0D0)                                         K400....
      DO 550 I=1,NN                                                     K410....
  550 UM2(I)=UM1(I)                                                     K420....
      GOTO 1000                                                         K430....
C                                                                       K440....
C.....INPUT INITIAL CONDITIONS FOR COLD START (UNIT-55 DATA)            K450....
  620 READ(55,510) TSTART                                               K460....
      READ(55,510) (PVEC(I),I=1,NN)                                     K470....
      READ(55,510) (UVEC(I),I=1,NN)                                     K480....
C.....START-UP WITH NO PROJECTIONS BY SETTING BDELP=BDELU=1.D-16        K490....
C     IN PROJECTION FORMULAE FOUND IN SUBROUTINE SUTRA.                 K500....
      DELTP=DELT*1.D16                                                  K510....
      DELTU=DELT*1.D16                                                  K520....
C.....INITIALIZE SPECIFIED TIME-VARYING PRESSURES TO INITIAL PRESSURE   K530....
C     VALUES FOR START-UP CALCULATION OF INFLOWS OR OUTFLOWS            K540....
C     (SET QPLITR=0)                                                    K550....
      IF(IPBCT) 680,740,740                                             K560....
  680 DO 730 IP=1,NPBC                                                  K570....
      I=IPBC(IP)                                                        K580....
      IF(I) 700,700,730                                                 K590....
  700 PBC(IP)=PVEC(-I)                                                  K600....
```

```
  730 CONTINUE                                                          K610....
C.....INITIALIZE P, U, AND CONSISTENT DENSITY                           K620....
  740 DO 800 I=1,NN                                                     K630....
      PM1(I)=PVEC(I)                                                    K640....
      UM1(I)=UVEC(I)                                                    K650....
      UM2(I)=UVEC(I)                                                    K660....
      RCIT(I)=RHOW0+DRWDU*(UVEC(I)-URHOW0)                              K670....
  800 CONTINUE                                                          K680....
C.....INITIALIZE SATURATION, SW(I)                                      K690....
      CALL ZERO(SW,NN,1.0D0)                                            K700....
      CALL ZERO(DSWDP,NN,0.0D0)                                         K710....
      IF(IUNSAT.NE.1) GOTO 990                                          K720....
      IUNSAT=3                                                          K730....
      DO 900 I=1,NN                                                     K740....
  900 IF(PVEC(I).LT.0) CALL UNSAT(SW(I),DSWDP(I),RELK,PVEC(I))          K750....
  990 CONTINUE                                                          K760....
      CALL ZERO(CS1,NN,CS)                                              K770....
C     CALL ZERO(CS2,NN,0.0D0)                                           K780....
C     CALL ZERO(CS3,NN,0.0D0)                                           K790....
      CALL ZERO(SL,NN,0.0D0)                                            K800....
      CALL ZERO(SR,NN,0.0D0)                                            K810....
 1000 CONTINUE                                                          K820....
C                                                                       K830....
C.....SET STARTING TIME OF SIMULATION CLOCK, TSEC                       K840....
      TSEC=TSTART                                                       K850....
C                                                                       K860....
C                                                                       K870....
      RETURN                                                            K880....
      END                                                               K890....
```

```
C     SUBROUTINE        P  R  I  S  O  L        SUTRA - VERSION 1284-2D L10.....


C     SUBROUTINE        P  R  I  S  O  L        SUTRA - VERSION 1284-2D L10.....
C                                                                       L20.....
C *** PURPOSE :                                                         L30.....
C ***  TO PRINT PRESSURE AND TEMPERATURE OR CONCENTRATION               L40.....
C ***  SOLUTIONS AND TO OUTPUT INFORMATION ON TIME STEP, ITERATIONS,     L50.....
C ***  SATURATIONS, AND FLUID VELOCITIES.                               L60.....
C                                                                       L70.....
      SUBROUTINE PRISOL(ML,ISTOP,IGOI,PVEC,UVEC,VMAG,VANG,SW)           L80.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               L90.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            L100....
     1   NSOP,NSOU,NBCN                                                 L110....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        L120....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  L130....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        L140....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        L150....
      COMMON/ITERAT/ RPM,RPMAX,RUM,RUMAX,ITER,ITRMAX,IPWORS,IUWORS      L160....
      COMMON/KPRINT/ KCORD,KELINE,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG       L170....
      DIMENSION PVEC(NN),UVEC(NN),VMAG(NE),VANG(NE),SW(NN)              L180....
C                                                                       L190....
C.....OUTPUT MAJOR HEADINGS FOR CURRENT TIME STEP                       L200....
      IF(IT.GT.0.OR.ISSFLO.EQ.2.OR.ISSTRA.EQ.1) GOTO 100                L210....
      WRITE(6,60)                                                       L220....
   60 FORMAT(1H1////11X,'I N I T I A L   C O N D I T I O N S',          L230....
     1          /11X,'_____________________________________')           L240....
      IF(IREAD.EQ.-1) WRITE(6,65)                                       L250....
   65 FORMAT(//11X,'INITIAL CONDITIONS RETRIEVED FROM STORAGE ',        L260....
     1   'ON UNIT 55.')                                                 L270....
      GOTO 500                                                          L280....
C                                                                       L290....
  100 IF(IGOI.NE.0.AND.ISTOP.EQ.0) WRITE(6,150) ITER,IT                 L300....
  150 FORMAT(/////////11X,'ITERATION ',I3,' SOLUTION FOR TIME STEP ',I4) L310....
C                                                                       L320....
      IF(ISTOP.EQ.-1) WRITE(6,250) IT,ITER                              L330....
  250 FORMAT(1H1//11X,'SOLUTION FOR TIME STEP ',I4,                     L340....
     1   ' NOT CONVERGED AFTER ',I3,' ITERATIONS.')                     L350....
C                                                                       L360....
      IF(ISTOP.GE.0) WRITE(6,350) IT                                    L370....
  350 FORMAT(1H1//11X,'RESULTS FOR TIME STEP ',I4/                      L380....
     1   11X,'_______ ___ ____ ____ ____')                              L390....
      IF(ITRMAX.EQ.1) GOTO 500                                          L400....
      IF(ISTOP.GE.0.AND.IT.GT.0) WRITE(6,355) ITER                      L410....
      IF(IT.EQ.0.AND.ISTOP.GE.0.AND.ISSFLO.EQ.2) WRITE(6,355) ITER      L420....
  355 FORMAT(11X,'(AFTER ',I3,' ITERATIONS) :')                         L430....
      WRITE(6,450) RPM,IPWORS,RUM,IUWORS                                L440....
  450 FORMAT(//11X,'MAXIMUM P CHANGE FROM PREVIOUS ITERATION ',         L450....
     1   1PD14.5,' AT NODE ',I5/11X,'MAXIMUM U CHANGE FROM PREVIOUS ',  L460....
     2   'ITERATION ',1PD14.5,' AT NODE ',I5)                           L470....
C                                                                       L480....
  500 IF(IT.EQ.0.AND.ISSFLO.EQ.2) GOTO 680                              L490....
      IF(ISSTRA.EQ.1) GOTO 800                                          L500....
      WRITE(6,550) DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,                     L510....
     1   TMONTH,TYEAR                                                   L520....
  550 FORMAT(////11X,'TIME INCREMENT :',T27,1PD15.4,' SECONDS'//11X,     L530....
     1   'ELAPSED TIME :',T27,1PD15.4,' SECONDS'/T27,1PD15.4,' MINUTES'L540....
     2   /T27,1PD15.4,' HOURS'/T27,1PD15.4,' DAYS'/T27,1PD15.4,' WEEKS'/L550....
     3   T27,1PD15.4,' MONTHS'/T27,1PD15.4,' YEARS')                    L560....
C                                                                       L570....
C.....OUTPUT PRESSURES FOR TRANSIENT FLOW SOLUTION (AND POSSIBLY,       L580....
C        SATURATION AND VELOCITY)                                       L590....
      IF(ML.EQ.2.AND.ISTOP.GE.0) GOTO 700                               L600....
```

```
C     SUBROUTINE        P  R  I  S  O  L        SUTRA - VERSION 1284-2D L10.....


      IF(ISSFLO.GT.0) GOTO 700                                          L610....
      WRITE(6,650) (I,PVEC(I),I=1,NN)                                   L620....
  650 FORMAT(///11X,'P  R  E  S  S  U  R  E'//8X,6('NODE',17X)/         L630....
     1   (7X,6(1X,I4,1X,1PD15.8)))                                      L640....
      IF(IUNSAT.NE.0) WRITE(6,651) (I,SW(I),I=1,NN)                     L650....
  651 FORMAT(///11X,'S  A  T  U  R  A  T  I  O  N'//8X,6('NODE',17X)/   L660....
     1   (7X,6(1X,I4,1X,1PD15.8)))                                      L670....
      IF(KVEL.EQ.1.AND.IT.GT.0) WRITE( 6,655) (L,VMAG(L),L=1,NE)        L680....
      IF(KVEL.EQ.1.AND.IT.GT.0) WRITE( 6,656) (L,VANG(L),L=1,NE)        L690....
  655 FORMAT(///11X,'F  L  U  I  D     V  E  L  O  C  I  T  Y'//        L700....
     1   11X,'M A G N I T U D E   AT CENTROID OF ELEMENT'//             L710....
     2   5X,6('ELEMENT',14X)/(7X,6(1X,I4,1X,1PD15.8)))                  L720....
  656 FORMAT(///11X,'F  L  U  I  D     V  E  L  O  C  I  T  Y'//        L730....
     1   11X,'A N G L E   IN DEGREES FROM +X-AXIS TO FLOW DIRECTION ',  L740....
     2   'AT CENTROID OF ELEMENT'//                                     L750....
     3   5X,6('ELEMENT',14X)/(7X,6(1X,I4,1X,1PD15.8)))                  L760....
      GOTO 700                                                          L770....
C                                                                       L780....
C.....OUTPUT PRESSURES FOR STEADY-STATE FLOW SOLUTION                   L790....
  680 WRITE(6,690) (I,PVEC(I),I=1,NN)                                   L800....
  690 FORMAT(///11X,'S  T  E  A  D  Y  -  S  T  A  T  E     P  R  E  S',L810....
     1   '  S  U  R  E'//8X,6('NODE',17X)/(7X,6(1X,I4,1X,1PD15.8)))     L820....
      IF(IUNSAT.NE.0) WRITE(6,651) (I,SW(I),I=1,NN)                     L830....
      GOTO 1000                                                         L840....
C                                                                       L850....
C.....OUTPUT CONCENTRATIONS OR TEMPERATURES FOR                         L860....
C         TRANSIENT TRANSPORT SOLUTION                                  L870....
  700 IF(ML.EQ.1.AND.ISTOP.GE.0) GOTO 1000                              L880....
      IF(ME) 720,720,730                                                L890....
  720 WRITE(6,725) (I,UVEC(I),I=1,NN)                                   L900....
  725 FORMAT(///11X,'C  O  N  C  E  N  T  R  A  T  I  O  N'//8X,        L910....
     1   6('NODE',17X)/(7X,6(1X,I4,1X,1PD15.8)))                        L920....
      GOTO 900                                                          L930....
  730 WRITE(6,735) (I,UVEC(I),I=1,NN)                                   L940....
  735 FORMAT(///11X,'T  E  M  P  E  R  A  T  U  R  E'//3X,6('NODE',17X)/L950....
     1   (7X,6(1X,I4,1X,F15.9)))                                        L960....
      GOTO 900                                                          L970....
C                                                                       L980....
C.....OUTPUT CONCENTRATIONS OR TEMPERATURES FOR                         L990....
C         STEADY-STATE TRANSPORT SOLUTION                               L1000...
  800 IF(ME) 820,820,830                                                L1010...
  820 WRITE(6,825) (I,UVEC(I),I=1,NN)                                   L1020...
  825 FORMAT(///11X,'S  T  E  A  D  Y  -  S  T  A  T  E     C  O  N  C', L1030...
     1   '  E  N  T  R  A  T  I  O  N'//3X,6('NODE',17X)/                L1040...
     2   (7X,6(1X,I4,1X,1PD15.8)))                                      L1050...
      GOTO 900                                                          L1060...
  830 WRITE(6,835) (I,UVEC(I),I=1,NN)                                   L1070...
  835 FORMAT(///11X,'S  T  E  A  D  Y  -  S  T  A  T  E     T  E  M  P', L1080...
     1   '  E  R  A  T  U  R  E'//8X,6('NODE',17X)/                     L1090...
     2   (7X,6(1X,I4,1X,F15.9)))                                        L1100...
C                                                                       L1110...
C.....OUTPUT VELOCITIES FOR STEADY-STATE FLOW SOLUTION                  L1120...
  900 IF(ISSFLO.NE.2.OR.IT.NE.1.OR.KVEL.NE.1) GOTO 1000                 L1130...
      WRITE( 6,925) (L,VMAG(L),L=1,NE)                                  L1140...
      WRITE( 6,950) (L,VANG(L),L=1,NE)                                  L1150...
  925 FORMAT(///11X,'S  T  E  A  D  Y  -  S  T  A  T  E     ',         L1160...
     1   'F  L  U  I  D     V  E  L  O  C  I  T  Y'//                   L1170...
     2   11X,'M A G N I T U D E   AT CENTROID OF ELEMENT'//             L1180...
     3   5X,6('ELEMENT',14X)/(7X,6(1X,I4,1X,1PD15.8)))                  L1190...
  950 FORMAT(///11X,'S  T  E  A  D  Y  -  S  T  A  T  E     ',         L1200...
```

```
C     SUBROUTINE        P  R  I  S  O  L        SUTRA - VERSION 1284-2D L10.....


     1   'F  L  U  I  D     V  E  L  O  C  I  T  Y'//                    L1210...
     2   11X,'A N G L E   IN DEGREES FROM +X-AXIS TO FLOW DIRECTION ',  L1220...
     3   'AT CENTROID OF ELEMENT'//                                      L1230...
     4   5X,6('ELEMENT',14X)/(7X,6(1X,I4,1X,1PD15.8)))                   L1240...
C                                                                        L1250...
 1000 RETURN                                                             L1260...
C                                                                        L1270...
      END                                                                L1280...
```

```
C     SUBROUTINE        Z  E  R  O              SUTRA - VERSION 1284-2D M10.....


C     SUBROUTINE        Z  E  R  O              SUTRA - VERSION 1284-2D M10.....
C                                                                     M20.....
C *** PURPOSE :                                                       M30.....
C ***  TO FILL AN ARRAY WITH A CONSTANT VALUE.                        M40.....
C                                                                     M50.....
      SUBROUTINE ZERO(A,IADIM,FILL)                                   M60.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                             M70.....
      DIMENSION A(IADIM)                                              M80.....
C                                                                     M90.....
C.....FILL ARRAY A WITH VALUE IN VARIABLE 'FILL'                      M100....
      DO 10 I=1,IADIM                                                 M110....
   10 A(I)=FILL                                                       M120....
C                                                                     M130....
C                                                                     M140....
      RETURN                                                          M150....
      END                                                             M160....
```

```
C     SUBROUTINE        B  C  T  I  M  E        SUTRA - VERSION 1284-2D N10.....


C     SUBROUTINE        B  C  T  I  M  E        SUTRA - VERSION 1284-2D N10.....
C                                                                       N20.....
C *** PURPOSE :                                                         N30.....
C ***  USER-PROGRAMMED SUBROUTINE WHICH ALLOWS THE USER TO SPECIFY:      N40.....
C ***    (1) TIME-DEPENDENT SPECIFIED PRESSURES AND TIME-DEPENDENT       N50.....
C ***        CONCENTRATIONS OR TEMPERATURES OF INFLOWS AT THESE POINTS   N60.....
C ***    (2) TIME-DEPENDENT SPECIFIED CONCENTRATIONS OR TEMPERATURES     N70.....
C ***    (3) TIME-DEPENDENT FLUID SOURCES AND CONCENTRATIONS             N80.....
C ***        OR TEMPERATURES OF INFLOWS AT THESE POINTS                  N90.....
C ***    (4) TIME-DEPENDENT ENERGY OR SOLUTE MASS SOURCES                N100....
C                                                                       N110....
      SUBROUTINE BCTIME(IPBC,PBC,IUBC,UBC,QIN,UIN,QUIN,IQSOP,IQSOU,     N120....
     1   IPBCT,IUBCT,IQSOPT,IQSOUT)                                     N130....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               N140....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            N150....
     1   NSOP,NSOU,NBCN                                                 N160....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        N170....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        N180....
      DIMENSION IPBC(NBCN),PBC(NBCN),IUBC(NBCN),UBC(NBCN),              N190....
     1   QIN(NN),UIN(NN),QUIN(NN),IQSOP(NSOP),IQSOU(NSOU)               N200....
C                                                                       N210....
C.....DEFINITION OF REQUIRED VARIABLES                                  N220....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N230....
C     NN = EXACT NUMBER OF NODES IN MESH                                N240....
C     NPBC = EXACT NUMBER OF SPECIFIED PRESSURE NODES                   N250....
C     NUBC = EXACT NUMBER OF SPECIFIED CONCENTRATION                    N260....
C            OR TEMPERATURE NODES                                       N270....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N280....
C     IT = NUMBER OF CURRENT TIME STEP                                  N290....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N300....
C     TSEC = TIME AT END OF CURRENT TIME STEP IN SECONDS                N310....
C     TMIN = TIME AT END OF CURRENT TIME STEP IN MINUTES                N320....
C     THOUR = TIME AT END OF CURRENT TIME STEP IN HOURS                 N330....
C     TDAY = TIME AT END OF CURRENT TIME STEP IN DAYS                   N340....
C     TWEEK = TIME AT END OF CURRENT TIME STEP IN WEEKS                 N350....
C     TMONTH = TIME AT END OF CURRENT TIME STEP IN MONTHS               N360....
C     TYEAR = TIME AT END OF CURRENT TIME STEP IN YEARS                 N370....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N380....
C     PBC(IP) = SPECIFIED PRESSURE VALUE AT IP(TH) SPECIFIED            N390....
C               PRESSURE NODE                                           N400....
C     UBC(IP) = SPECIFIED CONCENTRATION OR TEMPERATURE VALUE OF ANY     N410....
C               INFLOW OCCURRING AT IP(TH) SPECIFIED PRESSURE NODE      N420....
C     IPBC(IP) = ACTUAL NODE NUMBER OF IP(TH) SPECIFIED PRESSURE NODE   N430....
C                [WHEN NODE NUMBER I=IPBC(IP) IS NEGATIVE (I<0),        N440....
C                VALUES MUST BE SPECIFIED FOR PBC AND UBC.]             N450....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N460....
C     UBC(IUP) = SPECIFIED CONCENTRATION OR TEMPERATURE VALUE AT        N470....
C                IU(TH) SPECIFIED CONCENTRATION OR TEMPERATURE NODE      N480....
C                (WHERE IUP=IU+NPBC)                                    N490....
C     IUBC(IUP) = ACTUAL NODE NUMBER OF IU(TH) SPECIFIED CONCENTRATION  N500....
C                 OR TEMPERATURE NODE (WHERE IUP=IU+NPBC)               N510....
C                 [WHEN NODE NUMBER I=IUBC(IU) IS NEGATIVE (I<0),       N520....
C                 A VALUE MUST BE SPECIFIED FOR UBC.]                   N530....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N540....
C     IQSOP(IQP) = NODE NUMBER OF IQP(TH) FLUID SOURCE NODE.            N550....
C                  [WHEN NODE NUMBER I=IQSOP(IQP) IS NEGATIVE (I<0),    N560....
C                  VALUES MUST BE SPECIFIED FOR QIN AND UIN.]           N570....
C     QIN(-I) = SPECIFIED FLUID SOURCE VALUE AT NODE (-I)               N580....
C     UIN(-I) = SPECIFIED CONCENTRATION OR TEMPERATURE VALUE OF ANY     N590....
C               INFLOW OCCURRING AT FLUID SOURCE NODE (-I)              N600....
```

```
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N610....
C     IQSOU(IQU) = NODE NUMBER OF IQU(TH) ENERGY OR                        N620....
C                  SOLUTE MASS SOURCE NODE                                 N630....
C                  [WHEN NODE NUMBER I=IQSOU(IQU) IS NEGATIVE (I<0),       N640....
C                  A VALUE MUST BE SPECIFIED FOR QUIN.]                    N650....
C     QUIN(-I) = SPECIFIED ENERGY OR SOLUTE MASS SOURCE VALUE              N660....
C                AT NODE (-I)                                              N670....
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . N680....
C                                                                         N690....
C                                                                         N700....
C.....NSOPI IS ACTUAL NUMBER OF FLUID SOURCE NODES                        N710....
      NSOPI=NSOP-1                                                        N720....
C.....NSOUI IS ACTUAL NUMBER OF ENERGY OR SOLUTE MASS SOURCE NODES        N730....
      NSOUI=NSOU-1                                                        N740....
C                                                                         N750....
C                                                                         N760....
C                                                                         N770....
C                                                                         N780....
C                                                                         N790....
C                                                                         N800....
      IF(IPBCT) 50,240,240                                                N810....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N820....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N830....
C.....SECTION (1):  SET TIME-DEPENDENT SPECIFIED PRESSURES OR             N840....
C     CONCENTRATIONS (TEMPERATURES) OF INFLOWS AT SPECIFIED               N850....
C     PRESSURE NODES                                                      N860....
C                                                                         N870....
   50 CONTINUE                                                            N880....
      DO 200 IP=1,NPBC                                                    N890....
      I=IPBC(IP)                                                          N900....
      IF(I) 100,200,200                                                   N910....
  100 CONTINUE                                                            N920....
C     NOTE : A FLOW AND TRANSPORT SOLUTION MUST OCCUR FOR ANY             N930....
C            TIME STEP IN WHICH PBC( ) CHANGES.                           N940....
C     PBC(IP) =  ((           ))                                          N950....
C     UBC(IP) =  ((           ))                                          N960....
  200 CONTINUE                                                            N970....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N980....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N990....
C                                                                         N1000...
C                                                                         N1010...
C                                                                         N1020...
C                                                                         N1030...
C                                                                         N1040...
C                                                                         N1050...
  240 IF(IUBCT) 250,440,440                                               N1060...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1070...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1080...
C.....SECTION (2):  SET TIME-DEPENDENT SPECIFIED                          N1090...
C     CONCENTRATIONS (TEMPERATURES)                                       N1100...
C                                                                         N1110...
  250 CONTINUE                                                            N1120...
      DO 400 IU=1,NUBC                                                    N1130...
      IUP=IU+NPBC                                                         N1140...
      I=IUBC(IUP)                                                         N1150...
      IF(I) 300,400,400                                                   N1160...
  300 CONTINUE                                                            N1170...
C     NOTE : A TRANSPORT SOLUTION MUST OCCUR FOR ANY TIME STEP            N1180...
C            IN WHICH UBC( ) CHANGES.  IN ADDITION, IF FLUID PROPERTIES N1190...
C            ARE SENSITIVE TO 'U' THEN A FLOW SOLUTION MUST OCCUR AS WELN1200...
```

```
C     SUBROUTINE        B  C  T  I  M  E        SUTRA - VERSION 1284-2D N10.....


C     UBC(IUP) =   ((          ))                                        N1210...
  400 CONTINUE                                                          N1220...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1230...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1240...
C                                                                       N1250...
C                                                                       N1260...
C                                                                       N1270...
C                                                                       N1280...
C                                                                       N1290...
C                                                                       N1300...
  440 IF(IQSOPT) 450,640,640                                            N1310...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1320...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1330...
C.....SECTION (3):  SET TIME-DEPENDENT FLUID SOURCES/SINKS,             N1340...
C      OR CONCENTRATIONS (TEMPERATURES) OF SOURCE FLUID                 N1350...
C                                                                       N1360...
  450 CONTINUE                                                          N1370...
      DO 600 IQP=1,NSOPI                                                N1380...
      I=IQSOP(IQP)                                                      N1390...
      IF(I) 500,600,600                                                 N1400...
  500 CONTINUE                                                          N1410...
C     NOTE : A FLOW AND TRANSPORT SOLUTION MUST OCCUR FOR ANY           N1420...
C            TIME STEP IN WHICH QIN( ) CHANGES.                         N1430...
C     QIN(-I) =   ((           ))                                       N1440...
C     NOTE : A TRANSPORT SOLUTION MUST OCCUR FOR ANY                    N1450...
C            TIME STEP IN WHICH UIN( ) CHANGES.                         N1460...
C     UIN(-I) =   ((           ))                                       N1470...
  600 CONTINUE                                                          N1480...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1490...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1500...
C                                                                       N1510...
C                                                                       N1520...
C                                                                       N1530...
C                                                                       N1540...
C                                                                       N1550...
C                                                                       N1560...
  640 IF(IQSOUT) 650,840,840                                            N1570...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1580...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1590...
C.....SECTION (4):  SET TIME-DEPENDENT SOURCES/SINKS                    N1600...
C     OF SOLUTE MASS OR ENERGY                                          N1610...
C                                                                       N1620...
  650 CONTINUE                                                          N1630...
      DO 800 IQU=1,NSOUI                                                N1640...
      I=IQSOU(IQU)                                                      N1650...
      IF(I) 700,800,800                                                 N1660...
  700 CONTINUE                                                          N1670...
C     NOTE : A TRANSPORT SOLUTION MUST OCCUR FOR ANY                    N1680...
C            TIME STEP IN WHICH QUIN( ) CHANGES.                        N1690...
C     QUIN(-I) =   ((           ))                                      N1700...
  800 CONTINUE                                                          N1710...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1720...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - N1730...
C                                                                       N1740...
C                                                                       N1750...
C                                                                       N1760...
C                                                                       N1770...
C                                                                       N1780...
C                                                                       N1790...
  840 CONTINUE                                                          N1800...
```

```
C     SUBROUTINE        B  C  T  I  M  E        SUTRA - VERSION 1284-2D N10.....

C                                                                       N1810...
      RETURN                                                            N1820...
      END                                                               N1830...
```

```
C     SUBROUTINE        A  D  S  O  R  B        SUTRA - VERSION 1284-2D 010.....


C     SUBROUTINE        A  D  S  O  R  B        SUTRA - VERSION 1284-2D 010.....
C                                                                       020.....
C *** PURPOSE :                                                         030.....
C ***  TO CALCULATE VALUES OF EQUILIBRIUM SORPTION PARAMETERS FOR        040.....
C ***  LINEAR, FREUNDLICH, AND LANGMUIR MODELS.                         050.....
C                                                                       060.....
      SUBROUTINE ADSORB(CS1,CS2,CS3,SL,SR,U)                            070.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               080.....
      CHARACTER*10 ADSMOD                                               090.....
      COMMON/MODSOR/ ADSMOD                                             0100....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            0110....
     1   NSOP,NSOU,NBCN                                                 0120....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,0130....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2       0140....
      DIMENSION CS1(NN),CS2(NN),CS3(NN),SL(NN),SR(NN),U(NN)             0150....
C                                                                       0160....
C.....NOTE THAT THE CONCENTRATION OF ADSORBATE, CS(I), IS GIVEN BY:     0170....
C     CS(I) = SL(I)*U(I) + SR(I)                                        0180....
C                                                                       0190....
C.....NO SORPTION                                                       0200....
      IF(ADSMOD.NE.'NONE      ') GOTO 450                               0210....
      DO 250 I=1,NN                                                     0220....
      CS1(I)=0.D0                                                       0230....
      CS2(I)=0.D0                                                       0240....
      CS3(I)=0.D0                                                       0250....
      SL(I)=0.D0                                                        0260....
      SR(I)=0.D0                                                        0270....
  250 CONTINUE                                                          0280....
      GOTO 2000                                                         0290....
C                                                                       0300....
C.....LINEAR SORPTION MODEL                                             0310....
  450 IF(ADSMOD.NE.'LINEAR    ') GOTO 700                               0320....
      DO 500 I=1,NN                                                     0330....
      CS1(I)=CHI1*RHOW0                                                 0340....
      CS2(I)=0.D0                                                       0350....
      CS3(I)=0.D0                                                       0360....
      SL(I)=CHI1*RHOW0                                                  0370....
      SR(I)=0.D0                                                        0380....
  500 CONTINUE                                                          0390....
      GOTO 2000                                                         0400....
C                                                                       0410....
C.....FREUNDLICH SORPTION MODEL                                         0420....
  700 IF(ADSMOD.NE.'FREUNDLICH') GOTO 950                               0430....
      CHCH=CHI1/CHI2                                                    0440....
      DCHI2=1.D0/CHI2                                                   0450....
      RH2=RHOW0**DCHI2                                                  0460....
      CHI2F=((1.D0-CHI2)/CHI2)                                          0470....
      CH12=CHI1**DCHI2                                                  0480....
      DO 750 I=1,NN                                                     0490....
      IF(U(I)) 720,720,730                                              0500....
  720 UCH=1.0D0                                                         0510....
      GOTO 740                                                          0520....
  730 UCH=U(I)**CHI2F                                                   0530....
  740 RU=RH2*UCH                                                        0540....
      CS1(I)=CHCH*RU                                                    0550....
      CS2(I)=0.D0                                                       0560....
      CS3(I)=0.D0                                                       0570....
      SL(I)=CH12*RU                                                     0580....
      SR(I)=0.D0                                                        0590....
  750 CONTINUE                                                          0600....
```

```
      GOTO 2000                                                          0610....
C                                                                        0620....
C.....LANGMUIR SORPTION MODEL                                            0630....
  950 IF(ADSMOD.NE.'LANGMUIR  ') GOTO 2000                               0640....
      DO 1000 I=1,NN                                                     0650....
      DD=1.D0+CHI2*RHOW0*U(I)                                            0660....
      CS1(I)=(CHI1*RHOW0)/(DD*DD)                                        0670....
      CS2(I)=0.D0                                                        0680....
      CS3(I)=0.D0                                                        0690....
      SL(I)=CS1(I)                                                       0700....
      SR(I)=CS1(I)*CHI2*RHOW0*U(I)*J(I)                                  0710....
 1000 CONTINUE                                                           0720....
C                                                                        0730....
 2000 RETURN                                                             0740....
      END                                                                0750....
```

```
C     SUBROUTINE        E  L  E  M  E  N        SUTRA - VERSION 1284-2D P10.....


C     SUBROUTINE        E  L  E  M  E  N        SUTRA - VERSION 1284-2D P10.....
C                                                                       P20.....
C *** PURPOSE :                                                         P30.....
C ***  TO CONTROL AND CARRY OUT ALL CALCULATIONS FOR EACH ELEMENT BY     P40.....
C ***  OBTAINING ELEMENT INFORMATION FROM THE BASIS FUNCTION ROUTINE,    P50.....
C ***  CARRYING OUT GAUSSIAN INTEGRATION OF FINITE ELEMENT INTEGRALS,   P60.....
C ***  AND SENDING RESULTS OF ELEMENT INTEGRATIONS TO GLOBAL ASSEMBLY   P70.....
C ***  ROUTINE.  ALSO CALCULATES VELOCITY AT EACH ELEMENT CENTROID FOR  P80.....
C ***  PRINTED OUTPUT.                                                  P90.....
C                                                                       P100....
      SUBROUTINE ELEMEN(ML,IN,X,Y,THICK,PITER,UITER,RCIT,RCITM1,POR,    P110....
     1   ALMAX,ALMIN,ATAVG,PERMXX,PERMXY,PERMYX,PERMYY,PANGLE,          P120....
     2   VMAG,VANG,VOL,PMAT,PVEC,UMAT,UVEC,GXSI,GETA,PVEL)              P130....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               P140....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            P150....
     1   NSOP,NSOU,NBCN                                                 P160....
      COMMON/TENSOR/ GRAVX,GRAVY                                        P170....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,P180....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2       P190....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        P200....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        P210....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        P220....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  P230....
      COMMON/KPRINT/ KNODAL,KELMNT,KINCID,KPLOTP,KPLOTU,KVEL,KBUDG      P240....
      DIMENSION IN(NIN),X(NN),Y(NN),THICK(NN),PITER(NN),                P250....
     1   UITER(NN),RCIT(NN),RCITM1(NN),POR(NN),PVEL(NN)                 P260....
      DIMENSION PERMXX(NE),PERMXY(NE),PERMYX(NE),PERMYY(NE),PANGLE(NE), P270....
     1   ALMAX(NE),ALMIN(NE),ATAVG(NE),VMAG(NE),VANG(NE),               P280....
     2   GXSI(NE,4),GETA(NE,4)                                          P290....
      DIMENSION VOL(NN),PMAT(NN,NBI),PVEC(NN),UMAT(NN,NBI),UVEC(NN)     P300....
      DIMENSION BFLOWE(4,4),DFLOWE(4),BTRANE(4,4),DTRANE(4,4),VOLE(4)   P310....
      DIMENSION F(4,4),W(4,4),DET(4),DFDXG(4,4),DFDYG(4,4),             P320....
     1   DWDXG(4,4),DWDYG(4,4)                                          P330....
      DIMENSION SWG(4),RHOG(4),VISCG(4),PORG(4),VXG(4),VYG(4),          P340....
     1   RELKG(4),RGXG(4),RGYG(4),VGMAG(4),THICKG(4)                    P350....
      DIMENSION RXXG(4),RXYG(4),RYXG(4),RYYG(4)                         P360....
      DIMENSION BXXG(4),BXYG(4),BYXG(4),BYYG(4),                        P370....
     1   EXG(4),EYG(4)                                                  P380....
      DIMENSION GXLOC(4),GYLOC(4)                                       P390....
      DATA GLOC/0.57735026918962D0/                                     P400....
      DATA INTIM/0/,ISTOP/0/,GXLOC/-1.D0,1.D0,1.D0,-1.D0/,              P410....
     1   GYLOC/-1.D0,-1.D0,1.D0,1.D0/                                   P420....
C                                                                       P430....
C.....DECIDE WHETHER TO CALCULATE CENTROID VELOCITIES ON THIS CALL      P440....
      IVPRNT=0                                                          P450....
      IF(MOD(IT,NPRINT).EQ.0.AND.ML.NE.2.AND.IT.NE.0) IVPRNT=1          P460....
      IF(IT.EQ.1) IVPRNT=1                                              P470....
      KVPRNT=IVPRNT+KVEL                                                P480....
C                                                                       P490....
C.....ON FIRST TIME STEP, PREPARE GRAVITY VECTOR COMPONENTS,            P500....
C        GXSI AND GETA, FOR CONSISTENT VELOCITIES,                      P510....
C        AND CHECK ELEMENT SHAPES                                       P520....
      IF(INTIM) 100,100,2000                                            P530....
  100 INTIM=1                                                           P540....
C.....LOOP THROUGH ALL ELEMENTS TO OBTAIN THE INVERSE JACOBIAN          P550....
C        AT EACH OF THE FOUR NODES IN EACH ELEMENT                      P560....
      DO 1000 L=1,NE                                                    P570....
       DO 500 IL=1,4                                                    P580....
        XLOC=GXLOC(IL)                                                  P590....
        YLOC=GYLOC(IL)                                                  P600....
```

```
        CALL BASIS2(0000,L,XLOC,YLOC,IN,X,Y,F(1,IL),W(1,IL),DET(IL),     P610....
     1     DFDXG(1,IL),DFDYG(1,IL),DWDXG(1,IL),DWDYG(1,IL),              P620....
     2     PITER,UITER,PVEL,POR,THICK,THICKG(IL),VXG(IL),VYG(IL),        P630....
     3     SWG(IL),RHOG(IL),VISCG(IL),PORG(IL),VGMAG(IL),RELKG(IL),      P640....
     4     PERMXX,PERMXY,PERMYX,PERMYY,CJ11,CJ12,CJ21,CJ22,              P650....
     5     GXSI,GETA,RCIT,RCITM1,RGXG(IL),RGYG(IL))                      P660....
        GXSI(L,IL)=CJ11*GRAVX+CJ12*GRAVY                                 P670....
        GETA(L,IL)=CJ21*GRAVX+CJ22*GRAVY                                 P680....
C.....CHECK FOR NEGATIVE- OR ZERO-AREA ERRORS IN ELEMENT SHAPES          P690....
        IF(DET(IL)) 200,200,500                                          P700....
  200   ISTOP=ISTOP+1                                                    P710....
        WRITE(6,400) IN((L-1)*4+IL),L,DET(IL)                            P720....
  400   FORMAT(11X,'THE DETERMINANT OF THE JACOBIAN AT GAUSS POINT ',I4,P730....
     1     ' IN ELEMENT ',I4,' IS NEGATIVE OR ZERO, ',1PE15.7)           P740....
  500  CONTINUE                                                          P750....
 1000 CONTINUE                                                          P760....
C                                                                       P770....
      IF(ISTOP.EQ.0) GOTO 2000                                          P780....
      WRITE(6,1500)                                                     P790....
 1500 FORMAT(///////11X,'SOME ELEMENTS HAVE INCORRECT GEOMETRY.'        P800....
     1   //11X,'PLEASE CHECK THE NODE COORDINATES AND ',                P810....
     2   'INCIDENCE LIST, MAKE CORRECTIONS, AND THEN RERUN.'////////    P820....
     3   11X,'S I M U L A T I O N   H A L T E D'/                       P830....
     4   11X,'_____________________   ___________')                     P840....
      ENDFILE(6)                                                        P850....
      STOP                                                              P860....
C                                                                       P870....
C.....LOOP THROUGH ALL ELEMENTS TO CARRY OUT SPATIAL INTEGRATION        P880....
C       OF FLUX TERMS IN P AND/CR U EQUATIONS                           P890....
 2000 IF(IUNSAT.NE.0) IUNSAT=2                                          P900....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P910....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P920....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P930....
      DO 9999 L=1,NE                                                    P940....
       XIX=-1.D0                                                        P950....
       YIY=-1.D0                                                        P960....
       KG=0                                                             P970....
C.....OBTAIN BASIS FUNCTION AND RELATED INFORMATION AT EACH OF          P980....
C       FOUR GAUSS POINTS IN THE ELEMENT                                P990....
       DO 2200 IYL=1,2                                                  P1000...
        DO 2100 IXL=1,2                                                 P1010...
         KG=KG+1                                                        P1020...
         XLOC=XIX*GLOC                                                  P1030...
         YLOC=YIY*GLOC                                                  P1040...
       CALL BASIS2(0001,L,XLOC,YLOC,IN,X,Y,F(1,KG),W(1,KG),DET(KG),     P1050...
     1     DFDXG(1,KG),DFDYG(1,KG),DWDXG(1,KG),DWDYG(1,KG),              P1060...
     2     PITER,UITER,PVEL,POR,THICK,THICKG(KG),VXG(KG),VYG(KG),        P1070...
     3     SWG(KG),RHOG(KG),VISCG(KG),PORG(KG),VGMAG(KG),RELKG(KG),      P1080...
     4     PERMXX,PERMXY,PERMYX,PERMYY,CJ11,CJ12,CJ21,CJ22,              P1090...
     5     GXSI,GETA,RCIT,RCITM1,RGXG(KG),RGYG(KG))                      P1100...
 2100    XIX=-XIX                                                       P1110...
 2200   YIY=-YIY                                                        P1120...
C                                                                       P1130...
C.....CALCULATE VELOCITY AT ELEMENT CENTROID WHEN REQUIRED              P1140...
       IF(KVPRNT-2) 3000,2300,3000                                      P1150...
 2300  AXSUM=0.0D0                                                      P1160...
       AYSUM=0.0D0                                                      P1170...
       DO 2400 KG=1,4                                                   P1180...
        AXSUM=AXSUM+VXG(KG)                                             P1190...
 2400   AYSUM=AYSUM+VYG(KG)                                             P1200...
```

```
      VMAG(L)=DSQRT(AXSUM*AXSUM+AYSUM*AYSUM)/4.0D0                          P1210...
      IF(AXSUM) 2500,2700,2800                                              P1220...
 2500 AYX=AYSUM/AXSUM                                                       P1230...
      VANG(L)=DATAN(AYX)/1.745329D-2                                        P1240...
      IF(AYSUM.LT.0.0D0) GOTO 2600                                          P1250...
      VANG(L)=VANG(L)+180.0D0                                               P1260...
      GOTO 3000                                                             P1270...
 2600 VANG(L)=VANG(L)-180.0D0                                               P1280...
      GOTO 3000                                                             P1290...
 2700 VANG(L)=90.0D0                                                        P1300...
      IF(AYSUM.LT.0.0D0) VANG(L)=-90.0D0                                    P1310...
      GOTO 3000                                                             P1320...
 2800 AYX=AYSUM/AXSUM                                                       P1330...
      VANG(L)=DATAN(AYX)/1.745329D-2                                        P1340...
C                                                                           P1350...
C.....INCLUDE MESH THICKNESS IN NUMERICAL INTEGRATION                       P1360...
 3000 DO 3300 KG=1,4                                                        P1370...
 3300  DET(KG)=THICKG(KG)*DET(KG)                                           P1380...
C                                                                           P1390...
C.....CALCULATE PARAMETERS FOR FLUID MASS BALANCE AT GAUSS POINTS           P1400...
      IF(ML-1) 3400,3400,6100                                               P1410...
 3400 SWTEST=0.D0                                                           P1420...
      DO 4000 KG=1,4                                                        P1430...
       SWTEST=SWTEST+SWG(KG)                                                P1440...
       ROMG=RHOG(KG)*RELKG(KG)/VISCG(KG)                                    P1450...
       RXXG(KG)=PERMXX(L)*ROMG                                              P1460...
       RXYG(KG)=PERMXY(L)*ROMG                                              P1470...
       RYXG(KG)=PERMYX(L)*ROMG                                              P1480...
       RYYG(KG)=PERMYY(L)*ROMG                                              P1490...
 4000  CONTINUE                                                             P1500...
C                                                                           P1510...
C.....INTEGRATE FLUID MASS BALANCE IN AN UNSATURATED ELEMENT                P1520...
C       USING ASYMMETRIC WEIGHTING FUNCTIONS                                P1530...
      IF(UP.LE.1.0D-6) GOTO 5200                                            P1540...
      IF(SWTEST-3.999D0) 4200,5200,5200                                     P1550...
 4200 DO 5000 I=1,4                                                         P1560...
       DF=0.D0                                                              P1570...
       VO=0.D0                                                              P1580...
       DO 4400 KG=1,4                                                       P1590...
        VO=VO+F(I,KG)*DET(KG)                                               P1600...
 4400   DF=DF+((RXXG(KG)*RGXG(KG)+RXYG(KG)*RGYG(KG))                       P1610...
     1          *DWDXG(I,KG)                                               P1620...
     2        + (RYXG(KG)*RGXG(KG)+RYYG(KG)*RGYG(KG))                      P1630...
     3          *DWDYG(I,KG))*DET(KG)                                      P1640...
       DO 4800 J=1,4                                                        P1650...
        BF=0.D0                                                             P1660...
        DO 4600 KG=1,4                                                      P1670...
 4600    BF=BF+((RXXG(KG)*DFDXG(J,KG)+RXYG(KG)*DFDYG(J,KG))*DWDXG(I,KG))   P1680...
     2        +(RYXG(KG)*DFDXG(J,KG)+RYYG(KG)*DFDYG(J,KG))*DWDYG(I,KG))    P1690...
     3          *DET(KG)                                                   P1700...
 4800   BFLOWE(I,J)=BF                                                      P1710...
       VOLE(I)=VO                                                           P1720...
 5000  DFLOWE(I)=DF                                                        P1730...
      GOTO 6200                                                             P1740...
C                                                                           P1750...
C.....INTEGRATE FLUID MASS BALANCE IN A SATURATED OR UNSATURATED            P1760...
C       ELEMENT USING SYMMETRIC WEIGHTING FUNCTIONS                         P1770...
 5200 DO 6000 I=1,4                                                         P1780...
       DF=0.D0                                                              P1790...
       VO=0.D0                                                              P1800...
```

```
      DO 5400 KG=1,4                                                    P1810...
       VO=VO+F(I,KG)*DET(KG)                                            P1820...
 5400  DF=DF+((RXXG(KG)*RGXG(KG)+RXYG(KG)*RGYG(KG))*DFDXG(I,KG)        P1830...
     2       + (RYXG(KG)*RGXG(KG)+RYYG(KG)*RGYG(KG))*DFDYG(I,KG))      P1840...
     3         *DET(KG)                                                P1850...
      DO 5800 J=1,4                                                     P1860...
       BF=0.DO                                                          P1870...
       DO 5600 KG=1,4                                                   P1880...
 5600   BF=BF+((RXXG(KG)*DFDXG(J,KG)+RXYG(KG)*DFDYG(J,KG))*DFDXG(I,KG)P1890...
     1       + (RYXG(KG)*DFDXG(J,KG)+RYYG(KG)*DFDYG(J,KG))*DFDYG(I,KG))P1900...
     2         *DET(KG)                                                P1910...
 5800  BFLOWE(I,J)=BF                                                   P1920...
      VOLE(I)=VO                                                        P1930...
 6000  DFLOWE(I)=DF                                                     P1940...
 6200 CONTINUE                                                          P1950...
      IF(ML-1) 6100,9000,6100                                           P1960...
 6100 IF(NOUMAT.EQ.1) GOTO 9000                                         P1970...
C                                                                       P1980...
C                                                                       P1990...
C.....CALCULATE PARAMETERS FOR ENERGY BALANCE OR SOLUTE MASS BALANCE    P2000...
C       AT GAUSS POINTS                                                 P2010...
      DO 7000 KG=1,4                                                    P2020...
       ESWG=PORG(KG)*SWG(KG)                                            P2030...
       RHOCWG=RHOG(KG)*CW                                               P2040...
       ESRCG=ESWG*RHOCWG                                                P2050...
       IF(VGMAG(KG)) 6300,6300,6600                                     P2060...
 6300  EXG(KG)=0.DO0                                                    P2070...
       EYG(KG)=0.000                                                    P2080...
       DXXG=0.000                                                       P2090...
       DXYG=0.000                                                       P2100...
       DYXG=0.000                                                       P2110...
       DYYG=0.DO0                                                       P2120...
       GOTO 6900                                                        P2130...
 6600  EXG(KG)=ESRCG*VXG(KG)                                            P2140...
       EYG(KG)=ESRCG*VYG(KG)                                            P2150...
C                                                                       P2160...
C.....DISPERSIVITY MODEL FOR ANISOTROPIC MEDIA                          P2170...
C       WITH PRINCIPAL DISPERSIVITIES:  ALMAX,ALMIN, AND ATAVG          P2180...
       VANGG=1.57079632700                                              P2190...
       IF(VXG(KG)*VXG(KG).GT.0.DO) VANGG=DATAN(VYG(KG)/VXG(KG))         P2200...
       VKANGG=VANGG-PANGLE(L)                                           P2210...
       DCO=DCOS(VKANGG)                                                 P2220...
       DSI=DSIN(VKANGG)                                                 P2230...
C.....EFFECTIVE LONGITUDINAL DISPERSIVITY IN FLOW DIRECTION, ALEFF      P2240...
       ALEFF=0.000                                                      P2250...
       IF(ALMAX(L)+ALMIN(L)) 6800,6800,6700                             P2260...
 6700  ALEFF=ALMAX(L)*ALMIN(L)/(ALMIN(L)*DCO*DCO+ALMAX(L)*DSI*DSI)      P2270...
 6800  DLG=ALEFF*VGMAG(KG)                                              P2280...
       DTG=ATAVG(L)*VGMAG(KG)                                           P2290...
C                                                                       P2300...
       V2GMI=1.DO/(VGMAG(KG)*VGMAG(KG))                                 P2310...
       V2ILTG=V2GMI*(DLG-DTG)                                           P2320...
       VX2G=VXG(KG)*VXG(KG)                                             P2330...
       VY2G=VYG(KG)*VYG(KG)                                             P2340...
C.....DISPERSION TENSOR                                                 P2350...
       DXXG=V2GMI*(DLG*VX2G+DTG*VY2G)                                   P2360...
       DYYG=V2GMI*(DTG*VX2G+DLG*VY2G)                                   P2370...
       DXYG=V2ILTG*VXG(KG)*VYG(KG)                                      P2380...
       DYXG=DXYG                                                        P2390...
C                                                                       P2400...
```

```
C.....IN-PARALLEL CONDUCTIVITIES (DIFFUSIVITIES) FORMULA                    P2410...
 6900   ESE=ESRCG*SIGMAW+(1.D0-PORG(KG))*RHOCWG*SIGMAS                      P2420...
C.....ADD DIFFUSION AND DISPERSION TERMS TO TOTAL DISPERSION TENSOR         P2430...
        BXXG(KG)=ESRCG*DXXG+ESE                                             P2440...
        BXYG(KG)=ESRCG*DXYG                                                 P2450...
        BYXG(KG)=ESRCG*DYXG                                                 P2460...
 7000   BYYG(KG)=ESRCG*DYYG+ESE                                             P2470...
C                                                                           P2480...
C.....INTEGRATE SOLUTE MASS BALANCE OR ENERGY BALANCE                       P2490...
C         USING SYMMETRIC WEIGHTING FUNCTIONS FOR DISPERSION TERM AND       P2500...
C         USING EITHER SYMMETRIC OR ASYMMETRIC WEIGHTING FUNCTIONS          P2510...
C         FOR ADVECTION TERM                                                P2520...
        DO 8000 I=1,4                                                       P2530...
         DO 8000 J=1,4                                                      P2540...
         BT=0.D0                                                            P2550...
         DT=0.D0                                                            P2560...
         DO 7500 KG=1,4                                                     P2570...
          BT=BT+((BXXG(KG)*DFDXG(J,KG)+BXYG(KG)*DFDYG(J,KG))*DFDXG(I,KG))P2580...
     1          +(BYXG(KG)*DFDXG(J,KG)+BYYG(KG)*DFDYG(J,KG))*DFDYG(I,KG))P2590...
     2           *DET(KG)                                                   P2600...
 7500     DT=DT+(EXG(KG)*DFDXG(J,KG)+EYG(KG)*DFDYG(J,KG))                  P2610...
     1           *W(I,KG)*DET(KG)                                           P2620...
         BTRANE(I,J)=BT                                                     P2630...
 8000    DTRANE(I,J)=DT                                                     P2640...
 9000  CONTINUE                                                            P2650...
C                                                                           P2660...
C                                                                           P2670...
C.....SEND RESULTS OF INTEGRATIONS FOR THIS ELEMENT TO                      P2680...
C         GLOBAL ASSEMBLY ROUTINE                                           P2690...
 9999  CALL GLOBAN(L,ML,VOLE,BFLOWE,DFLOWE,BTRANE,DTRANE,                  P2700...
     1    IN,VOL,PMAT,PVEC,UMAT,UVEC)                                       P2710...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P2720...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P2730...
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - P2740...
C                                                                           P2750...
C                                                                           P2760...
       RETURN                                                               P2770...
       END                                                                  P2780...
```

```
C     SUBROUTINE        B  A  S  I  S  2        SUTRA - VERSION 1284-2D Q10.....


C     SUBROUTINE        B  A  S  I  S  2        SUTRA - VERSION 1284-2D Q10.....
C                                                                       Q20.....
C *** PURPOSE :                                                         Q30.....
C ***  TO CALCULATE VALUES OF BASIS AND WEIGHTING FUNCTIONS AND THEIR   Q40.....
C ***  DERIVATIVES, TRANSFORMATION MATRICES BETWEEN LOCAL AND GLOBAL    Q50.....
C ***  COORDINATES AND PARAMETER VALUES AT A SPECIFIED POINT IN A       Q60.....
C ***  QUADRILATERAL FINITE ELEMENT.                                    Q70.....
C                                                                       Q80.....
      SUBROUTINE BASIS2(ICALL,L,XLOC,YLOC,IN,X,Y,F,W,DET,               Q90.....
     1   DFDXG,DFDYG,DWDXG,DWDYG,PITER,UITER,PVEL,POR,THICK,THICKG,     Q100....
     2   VXG,VYG,SWG,RHOG,VISCG,PORG,VGMAG,RELKG,                       Q110....
     3   PERMXX,PERMXY,PERMYX,PERMYY,CJ11,CJ12,CJ21,CJ22,               Q120....
     4   GXSI,GETA,RCIT,RCITM1,RGXG,RGYG)                               Q130....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               Q140....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            Q150....
     1   NSOP,NSOU,NBCN                                                 Q160....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        Q170....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  Q180....
      COMMON/SATPAR/ PCENT,SWRES,PCRES,SSLOPE,SINCPT                    Q190....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,Q200....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2        Q210....
      COMMON/TENSOR/ GRAVX,GRAVY                                        Q220....
      DOUBLE PRECISION XLOC,YLOC                                        Q230....
      DIMENSION IN(NIN),X(NN),Y(NN),UITER(NN),PITER(NN),PVEL(NN),       Q240....
     1   POR(NN),PERMXX(NE),PERMXY(NE),PERMYX(NE),PERMYY(NE),THICK(NN)  Q250....
      DIMENSION GXSI(NE,4),GETA(NE,4),RCIT(NN),RCITM1(NN)               Q260....
      DIMENSION F(4),W(4),DFDXG(4),DFDYG(4),DWDXG(4),DWDYG(4)           Q270....
      DIMENSION FX(4),FY(4),AFX(4),AFY(4),                              Q280....
     1   DFDXL(4),DFDYL(4),DWDXL(4),DWDYL(4),                           Q290....
     2   XDW(4),YDW(4),XIIX(4),YIIY(4)                                  Q300....
      DATA XIIX/-1.D0,+1.D0,+1.D0,-1.D0/,                               Q310....
     1   YIIY/-1.D0,-1.D0,+1.D0,+1.D0/                                  Q320....
C                                                                       Q330....
C                                                                       Q340....
C.....AT THIS LOCATION IN LOCAL COORDINATES, (XLOC,YLOC),               Q350....
C        CALCULATE SYMMETRIC WEIGHTING FUNCTIONS, F(I),                 Q360....
C        SPACE DERIVATIVES, DFDXG(I) AND DFDYG(I), AND                  Q370....
C        DETERMINANT OF JACOBIAN, DET.                                  Q380....
C                                                                       Q390....
      XF1=1.D0-XLOC                                                     Q400....
      XF2=1.D0+XLOC                                                     Q410....
      YF1=1.D0-YLOC                                                     Q420....
      YF2=1.D0+YLOC                                                     Q430....
C                                                                       Q440....
C.....CALCULATE BASIS FUNCTION, F.                                      Q450....
      FX(1)=XF1                                                         Q460....
      FX(2)=XF2                                                         Q470....
      FX(3)=XF2                                                         Q480....
      FX(4)=XF1                                                         Q490....
      FY(1)=YF1                                                         Q500....
      FY(2)=YF1                                                         Q510....
      FY(3)=YF2                                                         Q520....
      FY(4)=YF2                                                         Q530....
      DO 10 I=1,4                                                       Q540....
   10 F(I)=0.250D0*FX(I)*FY(I)                                          Q550....
C                                                                       Q560....
C.....CALCULATE DERIVATIVES WITH RESPECT TO LOCAL COORDINATES.          Q570....
      DO 20 I=1,4                                                       Q580....
      DFDXL(I)=XIIX(I)*0.250D0*FY(I)                                    Q590....
   20 DFDYL(I)=YIIY(I)*0.250D0*FX(I)                                    Q600....
```

```
C     SUBROUTINE        B  A  S  I  S  2        SUTRA - VERSION 1284-2D Q10.....


C                                                                       Q610....
C.....CALCULATE ELEMENTS OF JACOBIAN MATRIX, CJ.                        Q620....
      CJ11=0.D0                                                         Q630....
      CJ12=0.D0                                                         Q640....
      CJ21=0.D0                                                         Q650....
      CJ22=0.D0                                                         Q660....
      DO 100 IL=1,4                                                     Q670....
      II=(L-1)*4+IL                                                     Q680....
      I=IN(II)                                                          Q690....
      CJ11=CJ11+DFDXL(IL)*X(I)                                          Q700....
      CJ12=CJ12+DFDXL(IL)*Y(I)                                          Q710....
      CJ21=CJ21+DFDYL(IL)*X(I)                                          Q720....
  100 CJ22=CJ22+DFDYL(IL)*Y(I)                                          Q730....
C                                                                       Q740....
C.....CALCULATE DETERMINANT OF JACOBIAN MATRIX.                         Q750....
      DET=CJ11*CJ22-CJ21*CJ12                                           Q760....
C                                                                       Q770....
C.....RETURN TO ELEMEN WITH JACOBIAN MATRIX ON FIRST TIME STEP.         Q780....
      IF(ICALL.EQ.0) RETURN                                             Q790....
C                                                                       Q800....
C.....CALCULATE ELEMENTS OF INVERSE JACOBIAN MATRIX, CIJ.               Q810....
      ODET=1.D0/DET                                                     Q820....
      CIJ11=+ODET*CJ22                                                  Q830....
      CIJ12=-ODET*CJ12                                                  Q840....
      CIJ21=-ODET*CJ21                                                  Q850....
      CIJ22=+ODET*CJ11                                                  Q860....
C                                                                       Q870....
C.....CALCULATE DERIVATIVES WITH RESPECT TO GLOBAL COORDINATES          Q880....
      DO 200 I=1,4                                                      Q890....
      DFDXG(I)=CIJ11*DFDXL(I)+CIJ12*DFDYL(I)                            Q900....
  200 DFDYG(I)=CIJ21*DFDXL(I)+CIJ22*DFDYL(I)                            Q910....
C                                                                       Q920....
C.....CALCULATE CONSISTENT COMPONENTS OF (RHO*GRAV) TERM IN LOCAL        Q930....
C        COORDINATES AT THIS LOCATION, (XLOC,YLOC)                      Q940....
      RGXL=0.D0                                                         Q950....
      RGYL=0.D0                                                         Q960....
      RGXLM1=0.D0                                                       Q970....
      RGYLM1=0.D0                                                       Q980....
      DO 800 IL=1,4                                                     Q990....
      II=(L-1)*4+IL                                                     Q1000...
      I=IN(II)                                                          Q1010...
      ADFDXL=DABS(DFDXL(IL))                                            Q1020...
      ADFDYL=DABS(DFDYL(IL))                                            Q1030...
      RGXL=RGXL+RCIT(I)*GXSI(L,IL)*ADFDXL                               Q1040...
      RGYL=RGYL+RCIT(I)*GETA(L,IL)*ADFDYL                               Q1050...
      RGXLM1=RGXLM1+RCITM1(I)*GXSI(L,IL)*ADFDXL                         Q1060...
      RGYLM1=RGYLM1+RCITM1(I)*GETA(L,IL)*ADFDYL                         Q1070...
  800 CONTINUE                                                          Q1080...
C                                                                       Q1090...
C.....TRANSFORM CONSISTENT COMPONENTS OF (RHO*GRAV) TERM TO             Q1100...
C        GLOBAL COORDINATES                                             Q1110...
      RGXG=CIJ11*RGXL+CIJ12*RGYL                                        Q1120...
      RGYG=CIJ21*RGXL+CIJ22*RGYL                                        Q1130...
      RGXGM1=CIJ11*RGXLM1+CIJ12*RGYLM1                                  Q1140...
      RGYGM1=CIJ21*RGXLM1+CIJ22*RGYLM1                                  Q1150...
C                                                                       Q1160...
C.....CALCULATE PARAMETER VALUES AT THIS LOCATION, (XLOC,YLOC)          Q1170...
C                                                                       Q1180...
      PITERG=0.D0                                                       Q1190...
      UITERG=0.D0                                                       Q1200...
```

```
      DPDXG=0.D0                                                        Q1210...
      DPDYG=0.D0                                                        Q1220...
      PORG=0.D0                                                         Q1230...
      THICKG=0.0D0                                                      Q1240...
      DO 1000 IL=1,4                                                    Q1250...
      II=(L-1)*4 +IL                                                    Q1260...
      I=IN(II)                                                          Q1270...
      DPDXG=DPDXG+PVEL(I)*DFDXG(IL)                                     Q1280...
      DPDYG=DPDYG+PVEL(I)*DFDYG(IL)                                     Q1290...
      PORG=PORG+POR(I)*F(IL)                                            Q1300...
      THICKG=THICKG+THICK(I)*F(IL)                                      Q1310...
      PITERG=PITERG+PITER(I)*F(IL)                                      Q1320...
      UITERG=UITERG+UITER(I)*F(IL)                                      Q1330...
 1000 CONTINUE                                                          Q1340...
C                                                                       Q1350...
C.....SET VALUES FOR DENSITY AND VISCOSITY                              Q1360...
C.....RHOG = FUNCTION(UITER)                                           Q1370...
      RHOG=RHOW0+DRWDU*(UITERG-URHOW0)                                  Q1380...
C.....VISCG = FUNCTION(UITER)                                          Q1390...
C        VISCOSITY IN UNITS OF VISC0*(KG/(M*SEC))                       Q1400...
      IF(ME) 1300,1300,1200                                             Q1410...
 1200 VISCG=VISC0*239.4D-7*(10.D0**(248.37D0/(UITERG+133.15D0)))        Q1420...
      GOTO 1400                                                         Q1430...
C.....FOR SOLUTE TRANSPORT... VISCG IS TAKEN TO BE CONSTANT             Q1440...
 1300 VISCG=VISC0                                                       Q1450...
 1400 CONTINUE                                                          Q1460...
C                                                                       Q1470...
C.....SET UNSATURATED FLOW PARAMETERS SWG AND RELKG                     Q1480...
      IF(IUNSAT-2) 1600,1500,1600                                       Q1490...
 1500 IF(PITERG) 1550,1600,1600                                         Q1500...
 1550 CALL UNSAT(SWG,DSWDPG,RELKG,PITERG)                               Q1510...
      GOTO 1700                                                         Q1520...
 1600 SWG=1.0D0                                                         Q1530...
      RELKG=1.0D0                                                       Q1540...
 1700 CONTINUE                                                          Q1550...
C                                                                       Q1560...
C.....CALCULATE CONSISTENT FLUID VELOCITIES WITH RESPECT TO GLOBAL      Q1570...
C        COORDINATES, VXG, VYG, AND VGMAG, AT THIS LOCATION, (XLOC,YLOC)Q1580...
      DENOM=1.D0/(PORG*SWG*VISCG)                                       Q1590...
      PGX=DPDXG-RGXGM1                                                  Q1600...
      PGY=DPDYG-RGYGM1                                                  Q1610...
C.....ZERO OUT RANDOM BOUYANT DRIVING FORCES DUE TO DIFFERENCING        Q1620...
C.....  NUMBERS PAST PRECISION LIMIT                                    Q1630...
C.....  MINIMUM DRIVING FORCE IS 1.D-10 OF PRESSURE GRADIENT           Q1640...
C.....  (THIS VALUE MAY BE CHANGED DEPENDING ON MACHINE PRECISION)      Q1650...
      IF(DPDXG) 1720,1730,1720                                          Q1660...
 1720 IF(DABS(PGX/DPDXG)-1.0D-10) 1725,1725,1730                        Q1670...
 1725 PGX=0.0D0                                                         Q1680...
 1730 IF(DPDYG) 1750,1760,1750                                          Q1690...
 1750 IF(DABS(PGY/DPDYG)-1.0D-10) 1755,1755,1760                        Q1700...
 1755 PGY=0.0D0                                                         Q1710...
 1760 VXG=-DENOM*(PERMXX(L)*PGX+PERMXY(L)*PGY)*RELKG                    Q1720...
      VYG=-DENOM*(PERMYX(L)*PGX+PERMYY(L)*PGY)*RELKG                    Q1730...
      VXG2=VXG*VXG                                                      Q1740...
      VYG2=VYG*VYG                                                      Q1750...
      VGMAG=DSQRT(VXG2+VYG2)                                            Q1760...
C                                                                       Q1770...
C.....AT THIS POINT IN LOCAL COORDINATES, (XLOC,YLOC),                  Q1780...
C       , CALCULATE ASYMMETRIC WEIGHTING FUNCTIONS, W(I),               Q1790...
C         AND SPACE DERIVATIVES, DWDXG(I) AND DWDYG(I).                 Q1800...
```

```
C                                                                       Q1810...
C.....ASYMMETRIC FUNCTIONS SIMPLIFY WHEN  UP=0.0                        Q1820...
      IF(UP.GT.1.0D-6.AND.NOUMAT.EQ.0) GOTO 1790                        Q1830...
      DO 1780 I=1,4                                                     Q1840...
      W(I)=F(I)                                                         Q1850...
      DWDXG(I)=DFDXG(I)                                                 Q1860...
      DWDYG(I)=DFDYG(I)                                                 Q1870...
 1780 CONTINUE                                                          Q1880...
C.....RETURN WHEN ONLY SYMMETRIC WEIGHTING FUNCTIONS ARE USED           Q1890...
      RETURN                                                            Q1900...
C                                                                       Q1910...
C.....CALCULATE FLUID VELOCITIES WITH RESPECT TO LOCAL COORDINATES,     Q1920...
C.....  VXL, VYL, AND VLMAG, AT THIS LOCATION, (XLOC,YLOC).            Q1930...
 1790 VXL=CIJ11*VXG+CIJ21*VYG                                          Q1940...
      VYL=CIJ12*VXG+CIJ22*VYG                                           Q1950...
      VLMAG=DSQRT(VXL*VXL+VYL*VYL)                                      Q1960...
C                                                                       Q1970...
      AA=0.0D0                                                          Q1980...
      BB=0.0D0                                                          Q1990...
      IF(VLMAG) 1900,1900,1800                                          Q2000...
 1800 AA=UP*VXL/VLMAG                                                   Q2010...
      BB=UP*VYL/VLMAG                                                   Q2020...
C                                                                       Q2030...
 1900 XIXI=.75D0*AA*XF1*XF2                                             Q2040...
      YIYI=.75D0*BB*YF1*YF2                                             Q2050...
      DO 2000 I=1,4                                                     Q2060...
      AFX(I)=.5D0*FX(I)+XIIX(I)*XIXI                                    Q2070...
 2000 AFY(I)=.5D0*FY(I)+YIIY(I)*YIYI                                    Q2080...
C                                                                       Q2090...
C.....CALCULATE ASYMMETRIC WEIGHTING FUNCTION, W.                       Q2100...
      DO 3000 I=1,4                                                     Q2110...
 3000 W(I)=AFX(I)*AFY(I)                                                Q2120...
C                                                                       Q2130...
      THAAX=0.5D0-1.5D0*AA*XLOC                                         Q2140...
      THBBY=0.5D0-1.5D0*BB*YLOC                                         Q2150...
      DO 4000 I=1,4                                                     Q2160...
      XDW(I)=XIIX(I)*THAAX                                              Q2170...
 4000 YDW(I)=YIIY(I)*THBBY                                              Q2180...
C                                                                       Q2190...
C.....CALCULATE DERIVATIVES WITH RESPECT TO LOCAL COORDINATES.          Q2200...
      DO 5000 I=1,4                                                     Q2210...
      DWDXL(I)=XDW(I)*AFY(I)                                            Q2220...
 5000 DWDYL(I)=YDW(I)*AFX(I)                                            Q2230...
C                                                                       Q2240...
C.....CALCULATE DERIVATIVES WITH RESPECT TO GLOBAL COORDINATES.         Q2250...
      DO 6000 I=1,4                                                     Q2260...
      DWDXG(I)=CIJ11*DWDXL(I)+CIJ12*DWDYL(I)                            Q2270...
 6000 DWDYG(I)=CIJ21*DWDXL(I)+CIJ22*DWDYL(I)                            Q2280...
C                                                                       Q2290...
C                                                                       Q2300...
      RETURN                                                            Q2310...
      END                                                               Q2320...
```

```
C     SUBROUTINE        U  N  S  A  T          SUTRA - VERSION 1284-2D R10.....

C     SUBROUTINE        U  N  S  A  T          SUTRA - VERSION 1284-2D R10.....
C                                                                     R20.....
C *** PURPOSE :                                                       R30.....
C ***  USER-PROGRAMMED SUBROUTINE GIVING:                             R40.....
C ***  (1)  SATURATION AS A FUNCTION OF PRESSURE ( SW(PRES) )         R50.....
C ***  (2)  DERIVATIVE OF SATURATION WITH RESPECT TO PRESSURE         R60.....
C ***       AS A FUNCTION OF EITHER PRESSURE OR SATURATION            R70.....
C ***       ( DSWDP(PRES), OR DSWDP(SW) )                            R80.....
C ***  (3)  RELATIVE PERMEABILITY AS A FUNCTION OF EITHER             R90.....
C ***       PRESSURE OR SATURATION ( REL(PRES) OR RELK(SW) )          R100....
C ***                                                                 R110....
C ***  CODE BETWEEN DASHED LINES MUST BE REPLACED TO GIVE THE         R120....
C ***  PARTICULAR UNSATURATED RELATIONSHIPS DESIRED.                  R130....
C                                                                     R140....
      SUBROUTINE UNSAT(SW,DSWDP,RELK,PRES)                            R150....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                             R160....
      COMMON/CONTRL/ GNJ,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,      R170....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                R180....
C                                                                     R190....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - --R200....
C     THREE PARAMETERS FOR UNSATURATED FLOW RELATIONSHIPS OF          R210....
C     VAN GENUCHTEN(1980)                                             R220....
C        RESIDUAL SATURATION, SWRES, GIVEN IN UNITS [L**0]            R230....
C        PARAMETER, AA, GIVEN IN INVERSE PRESSURE UNITS [m*(s**2)/kg] R240....
C        PARAMETER, VN, GIVEN IN UNITS [L**0]                         R250....
      DATA   SWRES/0.3000/,   AA/5.0D-5/,   VN/2.000/                 R260....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - --R270....
C                                                                     R280....
C                                                                     R290....
C                                                                     R300....
C                                                                     R310....
C                                                                     R320....
C                                                                     R330....
C*********************************************************************R340....
C*********************************************************************R350....
C.....SECTION (1):                                                    R360....
C     SW VS. PRES   (VALUE CALCULATED ON EACH CALL TO UNSAT)          R370....
C     CODING MUST GIVE A VALUE TO SATURATION, SW.                     R380....
C                                                                     R390....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R400....
C     THREE PARAMETER MODEL OF VAN GENUCHTEN(1980)                    R410....
      SWRM1=1.D0-SWRES                                                R420....
      AAPVN=1.D0+(AA*(-PRES))**VN                                     R430....
      VNF=(VN-1.D0)/VN                                                R440....
      AAPVNN=AAPVN**VNF                                               R450....
      S W    =    SWRES+SWRM1/AAPVNN                                  R460....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R470....
C*********************************************************************R480....
C*********************************************************************R490....
C                                                                     R500....
C                                                                     R510....
C                                                                     R520....
C                                                                     R530....
C                                                                     R540....
C                                                                     R550....
      IF(IUNSAT-2) 600,1200,1800                                      R560....
C*********************************************************************R570....
C*********************************************************************R580....
C.....SECTION (2):                                                    R590....
C     DSWDP VS. PRES, OR DSWDP VS. SW   (CALCULATED ONLY WHEN IUNSAT=1) R600....
```

```
C     SUBROUTINE        U  N  S  A  T          SUTRA - VERSION 1284-2D R10.....


C     CODING MUST GIVE A VALUE TO DERIVATIVE OF SATURATION WITH         R610....
C     RESPECT TO PRESSURE, DSWDP.                                       R620....
C                                                                       R630....
  600 CONTINUE                                                          R640....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R650....
      DNUM=AA*(VN-1.D0)*SWRM1*(AA*(-PRES))**(VN-1.D0)                   R660....
      DNOM=AAPVN*AAPVNN                                                 R670....
      D S W D P   =   DNUM/DNOM                                         R680....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R690....
      GOTO 1800                                                         R700....
C*************************************************************************R710....
C*************************************************************************R720....
C                                                                       R730....
C                                                                       R740....
C                                                                       R750....
C                                                                       R760....
C                                                                       R770....
C                                                                       R780....
C*************************************************************************R790....
C*************************************************************************R800....
C.....SECTION (3):                                                      R810....
C     RELK VS. P, OR RELK VS. SW   (CALCULATED ONLY WHEN IUNSAT=2)      R820....
C     CODING MUST GIVE A VALUE TO RELATIVE PERMEABILITY, RELK.          R830....
C                                                                       R840....
 1200 CONTINUE                                                          R850....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R860....
C     GENERAL RELATIVE PERMEABILITY MODEL FROM VAN GENUCHTEN(1980)      R870....
      SWSTAR=(SW-SWRES)/SWRM1                                           R880....
      R E L K   =   DSQRT(SWSTAR)*                                      R890....
     1                  (1.D0-(1.D0-SWSTAR**(1.D0/VNF))**(VNF))**2.D0  R900....
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - R910....
C                                                                       R920....
C*************************************************************************R930....
C*************************************************************************R940....
C                                                                       R950....
C                                                                       R960....
C                                                                       R970....
C                                                                       R980....
C                                                                       R990....
C                                                                       R1000...
 1800 RETURN                                                            R1010...
C                                                                       R1020...
      END                                                               R1030...
```

```
C     SUBROUTINE        G  L  O  B  A  N        SUTRA - VERSION 1284-2D S10.....


C     SUBROUTINE        G  L  O  B  A  N        SUTRA - VERSION 1284-2D S10.....
C                                                                       S20.....
C *** PURPOSE :                                                         S30.....
C ***  TO ASSEMBLE RESULTS OF ELEMENTWISE INTEGRATIONS INTO             S40.....
C ***  A GLOBAL BANDED MATRIX AND GLOBAL VECTOR FOR BOTH                S50.....
C ***  FLOW AND TRANSPORT EQUATIONS.                                    S60.....
C                                                                       S70.....
      SUBROUTINE GLOBAN(L,ML,VOLE,BFLOWE,DFLOWE,BTRANE,DTRANE,          S80.....
     1   IN,VOL,PMAT,PVEC,UMAT,UVEC)                                    S90.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               S100....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            S110....
     1   NSOP,NSOU,NBCN                                                 S120....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        S130....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  S140....
      DIMENSION BFLOWE(4,4),DFLOWE(4),BTRANE(4,4),DTRANE(4,4),VOLE(4)   S150....
      DIMENSION VOL(NN),PMAT(NN,NBI),PVEC(NN),UMAT(NN,NBI),UVEC(NN)     S160....
      DIMENSION IN(NIN)                                                 S170....
C                                                                       S180....
      N1=(L-1)*4+1                                                      S190....
      N4=N1+3                                                           S200....
C                                                                       S210....
C.....ADD RESULTS OF INTEGRATIONS OVER ELEMENT L TO GLOBAL              S220....
C        P-MATRIX AND P-VECTOR                                          S230....
      IF(ML-1) 50,50,150                                                S240....
   50 IE=0                                                              S250....
      DO 100 II=N1,N4                                                   S260....
      IE=IE+1                                                           S270....
      IB=IN(II)                                                         S280....
      VOL(IB)=VOL(IB)+VOLE(IE)                                          S290....
      PVEC(IB)=PVEC(IB)+DFLOWE(IE)                                      S300....
      JE=0                                                              S310....
      DO 100 JJ=N1,N4                                                   S320....
      JE=JE+1                                                           S330....
      JB=IN(JJ)-IB+NBHALF                                               S340....
  100 PMAT(IB,JB)=PMAT(IB,JB)+BFLOWE(IE,JE)                             S350....
      IF(ML-1) 150,300,150                                              S360....
C                                                                       S370....
C.....ADD RESULTS OF INTEGRATIONS OVER ELEMENT L TO GLOBAL              S380....
C        U-MATRIX                                                       S390....
  150 IF(NOUMAT.EQ.1) GOTO 300                                          S400....
      IE=0                                                              S410....
      DO 200 II=N1,N4                                                   S420....
      IE=IE+1                                                           S430....
      IB=IN(II)                                                         S440....
C.....POSITION FOR ADDITION TO U-VECTOR                                 S450....
C     UVEC(IB)=UVEC(IB)+ ((    ))                                       S460....
      JE=0                                                              S470....
      DO 200 JJ=N1,N4                                                   S480....
      JE=JE+1                                                           S490....
      JB=IN(JJ)-IB+NBHALF                                               S500....
  200 UMAT(IB,JB)=UMAT(IB,JB)+DTRANE(IE,JE)+BTRANE(IE,JE)               S510....
C                                                                       S520....
  300 CONTINUE                                                          S530....
C                                                                       S540....
C                                                                       S550....
      RETURN                                                            S560....
      END                                                               S570....
```

```
C     SUBROUTINE        N  O  D  A  L  B        SUTRA - VERSION 1284-2D T10.....



C     SUBROUTINE        N  O  D  A  L  B        SUTRA - VERSION 1284-2D T10.....
C                                                                       T20.....
C *** PURPOSE :                                                         T30.....
C ***  (1) TO CARRY OUT ALL CELLWISE CALCULATIONS AND TO ADD CELLWISE    T40.....
C ***      TERMS TO THE GLOBAL BANDED MATRIX AND GLOBAL VECTOR FOR      T50.....
C ***      BOTH FLOW AND TRANSPORT EQUATIONS.                           T60.....
C ***  (2) TO ADD FLUID SOURCE AND SOLUTE MASS OR ENERGY SOURCE TERMS   T70.....
C ***      TO THE MATRIX EQUATIONS.                                     T80.....
C                                                                       T90.....
      SUBROUTINE NODALB(ML,VOL,PMAT,PVEC,UMAT,UVEC,PITER,UITER,PM1,UM1, T100....
     1   UM2,POR,QIN,UIN,QUIN,CS1,CS2,CS3,SL,SR,SW,DSWDP,RHO,SOP)        T110....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               T120....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            T130....
     1   NSOP,NSOU,NBCN                                                 T140....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,        T150....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                        T160....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,T170....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2       T180....
      COMMON/SATPAR/ PCENT,SWRES,PCRES,SSLOPE,SINCPT                    T190....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        T200....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  T210....
      DIMENSION VOL(NN),PMAT(NN,NBI),PVEC(NN),UMAT(NN,NBI),UVEC(NN)     T220....
      DIMENSION PITER(NN),UITER(NN),PM1(NN),UM1(NN),UM2(NN),            T230....
     1   POR(NN),QIN(NN),UIN(NN),QUIN(NN),CS1(NN),CS2(NN),CS3(NN),      T240....
     2   SL(NN),SR(NN),SW(NN),RHO(NN),DSWDP(NN),SOP(NN)                 T250....
C                                                                       T260....
C                                                                       T270....
      IF(IUNSAT.NE.0) IUNSAT=1                                          T280....
C                                                                       T290....
C.....DO NOT UPDATE NODAL PARAMETERS ON A TIME STEP WHEN ONLY U IS      T300....
C     SOLVED FOR BY BACK SUBSTITUTION (IE: WHEN NOUMAT=1)               T310....
      IF(NOUMAT) 50,50,200                                              T320....
C.....SET UNSATURATED FLOW PARAMETERS AT NODES, SW(I) AND DSWDP(I)      T330....
   50 DO 120 I=1,NN                                                     T340....
      IF(IUNSAT-1) 120,100,120                                          T350....
  100 IF(PITER(I)) 110,120,120                                          T360....
  110 CALL UNSAT(SW(I),DSWDP(I),RELK,PITER(I))                          T370....
  120 CONTINUE                                                          T380....
C.....SET FLUID DENSITY AT NODES, RHO(I)                                T390....
C     RHO = F (UITER(I))                                                T400....
      DO 150 I=1,NN                                                     T410....
  150 RHO(I)=RHOW0+DRWDU*(UITER(I)-URHOW0)                              T420....
  200 CONTINUE                                                          T430....
C                                                                       T440....
      DO 1000 I=1,NN                                                    T450....
      SWRHON=SW(I)*RHO(I)                                               T460....
C                                                                       T470....
      IF(ML-1) 220,220,230                                              T480....
C                                                                       T490....
C.....CALCULATE CELLWISE TERMS FOR P EQUATION                           T500....
C.....FOR STEADY-STATE FLOW, ISSFLO=2; FOR TRANSIENT FLOW, ISSFLO=0     T510....
  220 AFLN=(1-ISSFLO/2)*                                                T520....
     1   (SWRHON*SOP(I)+POR(I)*RHO(I)*DSWDP(I))*VOL(I)/DELTP            T530....
      CFLN=POR(I)*SW(I)*DRWDU*VOL(I)                                    T540....
      DUDT=(1-ISSFLO/2)*(UM1(I)-UM2(I))/DLTUM1                          T550....
      CFLN=CFLN*DUDT                                                    T560....
C.....ADD CELLWISE TERMS AND FLUID SOURCES OR FLUXES TO P EQUATION      T570....
      PMAT(I,NBHALF) = PMAT(I,NBHALF) + AFLN                            T580....
      PVEC(I) = PVEC(I) - CFLN + AFLN*PM1(I) + QIN(I)                   T590....
C                                                                       T600....
```

```
C     SUBROUTINE        N  O  D  A  L  B        SUTRA - VERSION 1284-2D T10.....


      IF(ML-1) 230,1000,230                                             T610....
C                                                                       T620....
C.....CALCULATE CELLWISE TERMS FOR U-EQUATION                           T630....
  230 EPRS=(1.DO-POR(I))*RHOS                                           T640....
      ATRN=(1-ISSTRA)*(POR(I)*SWRHON*CW+EPRS*CS1(I))*VOL(I)/DELTU       T650....
      GTRN=POR(I)*SWRHON*PRODF1*VOL(I)                                  T660....
      GSV=EPRS*PRODS1*VOL(I)                                            T670....
      GSLTRN=GSV*SL(I)                                                  T680....
      GSRTRN=GSV*SR(I)                                                  T690....
      ETRN=(POR(I)*SWRHON*PRODF0+EPRS*PRODS0)*VOL(I)                    T700....
C.....CALCULATE SOURCES OF SOLUTE OR ENERGY CONTAINED IN                T710....
C     SOURCES OF FLUID (ZERO CONTRIBUTION FOR OUTFLOWING FLUID)         T720....
      QUR=0.0D0                                                         T730....
      QUL=0.0D0                                                         T740....
      IF(QIN(I)) 360,360,340                                            T750....
  340 QUL=-CW*QIN(I)                                                    T760....
      QUR=-QUL*UIN(I)                                                   T770....
C.....ADD CELLWISE TERMS, SOURCES OF SOLUTE OR ENERGY IN FLUID INFLOWS, T780....
C        AND PURE SOURCES OR FLUXES OF SOLUTE OR ENERGY TO U-EQUATION   T790....
  360 IF(NOUMAT) 370,370,380                                            T800....
  370 UMAT(I,NBHALF) = UMAT(I,NBHALF) + ATRN - GTRN - GSLTRN - QUL      T810....
  380 UVEC(I) = UVEC(I) + ATRN*UM1(I) + ETRN + GSRTRN + QUR + QUIN(I)   T820....
C                                                                       T830....
 1000 CONTINUE                                                          T840....
C                                                                       T850....
      RETURN                                                            T860....
      END                                                               T870....
```

```
C     SUBROUTINE        B  C  B                    SUTRA - VERSION 1284-2D U10.....


C     SUBROUTINE        B  C  B                    SUTRA - VERSION 1284-2D U10.....
C                                                                          U20.....
C *** PURPOSE :                                                            U30.....
C ***  TO IMPLEMENT SPECIFIED PRESSURE AND SPECIFIED TEMPERATURE OR         U40.....
C ***  CONCENTRATION CONDITIONS BY MODIFYING THE GLOBAL FLOW AND            U50.....
C ***  TRANSPORT MATRIX EQUATIONS.                                         U60.....
C                                                                          U70.....
      SUBROUTINE BCB(ML,PMAT,PVEC,UMAT,UVEC,IPBC,PBC,IUBC,UBC,QPLITR)      U80.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                                  U90.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,               U100....
     1   NSOP,NSOU,NBCN                                                    U110....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,           U120....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                           U130....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,   U140....
     1   RHOWO,URHOWO,VISCO,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2          U150....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,           U160....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                     U170....
      DIMENSION PMAT(NN,NBI),PVEC(NN),UMAT(NN,NBI),UVEC(NN),               U180....
     1   IPBC(NBCN),PBC(NBCN),IUBC(NBCN),UBC(NBCN),QPLITR(NBCN)            U190....
C                                                                          U200....
C                                                                          U210....
      IF(NPBC.EQ.0) GOTO 1050                                              U220....
C.....SPECIFIED P BOUNDARY CONDITIONS                                      U230....
      DO 1000 IP=1,NPBC                                                    U240....
      I=IABS(IPBC(IP))                                                     U250....
C                                                                          U260....
      IF(ML-1) 100,100,200                                                 U270....
C.....MODIFY EQUATION FOR P BY ADDING FLUID SOURCE AT SPECIFIED            U280....
C        PRESSURE NODE                                                     U290....
  100 GINL=-GNU                                                            U300....
      GINR=GNU*PBC(IP)                                                     U310....
      PMAT(I,NBHALF)=PMAT(I,NBHALF)-GINL                                   U320....
      PVEC(I)=PVEC(I)+GINR                                                 U330....
C                                                                          U340....
      IF(ML-1) 200,1000,200                                                U350....
C.....MODIFY EQUATION FOR U BY ADDING U SOURCE WHEN FLUID FLOWS IN         U360....
C        AT SPECIFIED PRESSURE NODE                                        U370....
  200 GUR=0.0D0                                                            U380....
      GUL=0.0D0                                                            U390....
      IF(QPLITR(IP)) 360,360,340                                           U400....
  340 GUL=-CW*QPLITR(IP)                                                   U410....
      GUR=-GUL*UBC(IP)                                                     U420....
  360 IF(NOUMAT) 370,370,380                                               U430....
  370 UMAT(I,NBHALF)=UMAT(I,NBHALF)-GUL                                    U440....
  380 UVEC(I)=UVEC(I)+GUR                                                  U450....
 1000 CONTINUE                                                             U460....
C                                                                          U470....
C                                                                          U480....
 1050 IF(ML-1) 1100,3000,1100                                              U490....
C.....SPECIFIED U BOUNDARY CONDITIONS                                      U500....
C        MODIFY U EQUATION AT SPECIFIED U NODE TO READ: U = UBC            U510....
 1100 IF(NUBC.EQ.0) GOTO 3000                                              U520....
      DO 2000 IU=1,NUBC                                                    U530....
      IUP=IU+NPBC                                                          U540....
      I=IABS(IUBC(IUP))                                                    U550....
      IF(NOUMAT) 1200,1200,2000                                            U560....
 1200 DO 1500 JB=1,NB                                                      U570....
 1500 JMAT(I,JB)=0.0D0                                                     U580....
      UMAT(I,NBHALF)=1.0D0                                                 U590....
 2000 UVEC(I)=UBC(IUP)                                                     U600....
```

```
C                                                                    U610....
 3000 CONTINUE                                                       U620....
C                                                                    U630....
C                                                                    U640....
      RETURN                                                         U650....
      END                                                            U660....
```

```
C     SUBROUTINE        P  I  N  C  H  B        SUTRA - VERSION 1284-2D V10.....


C     SUBROUTINE        P  I  N  C  H  B        SUTRA - VERSION 1284-2D V10.....
C                                                                       V20.....
C *** PURPOSE :                                                         V30.....
C ***  TO IMPLEMENT PINCH NODE CONDITIONS BY MODIFYING THE              V40.....
C ***  GLOBAL FLOW AND TRANSPORT MATRIX EQUATIONS.                      V50.....
C                                                                       V60.....
      SUBROUTINE PINCHB(ML,IPINCH,PMAT,PVEC,UMAT,UVEC)                  V70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               V80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,            V90.....
     1   NSOP,NSOU,NBCN                                                 V100....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,        V110....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT                  V120....
      DIMENSION IPINCH(NPINCH,3),PMAT(NN,NBI),PVEC(NN),                 V130....
     1   UMAT(NN,NBI),UVEC(NN)                                          V140....
C                                                                       V150....
C.....NPIN IS ACTUAL NUMBER OF PINCH NODES IN MESH                      V160....
      NPIN=NPINCH-1                                                     V170....
      DO 1000 IPIN=1,NPIN                                               V180....
C.....SET NUMBERS OF PINCH NODE AND NEIGHBOR NODES                      V190....
      I=IPINCH(IPIN,1)                                                  V200....
      ICOR1=IPINCH(IPIN,2)                                              V210....
      ICOR2=IPINCH(IPIN,3)                                              V220....
      JC1=ICOR1-I+NBHALF                                                V230....
      JC2=ICOR2-I+NBHALF                                                V240....
C                                                                       V250....
      IF(ML-1) 50,50,250                                                V260....
C.....ADJUST P EQUATION FOR PINCH NODE CONDITIONS                       V270....
   50 DO 100 JB=1,NB                                                    V280....
  100 PMAT(I,JB)=0.0D0                                                  V290....
      PVEC(I)=0.0D0                                                     V300....
      PMAT(I,NBHALF)=1.0D0                                              V310....
      PMAT(I,JC1)=-0.5D0                                                V320....
      PMAT(I,JC2)=-0.5D0                                                V330....
      IF(ML-1) 250,1000,250                                             V340....
C.....ADJUST U EQUATION FOR PINCH NODE CONDITIONS                       V350....
  250 IF(NOUMAT) 300,300,500                                            V360....
  300 DO 400 JB=1,NB                                                    V370....
  400 UMAT(I,JB)=0.0D0                                                  V380....
      UMAT(I,NBHALF)=1.0D0                                              V390....
      UMAT(I,JC1)=-0.5D0                                                V400....
      UMAT(I,JC2)=-0.5D0                                                V410....
  500 UVEC(I)=0.0D0                                                     V420....
C                                                                       V430....
 1000 CONTINUE                                                          V440....
C                                                                       V450....
C                                                                       V460....
      RETURN                                                            V470....
      END                                                               V480....
```

```
C     SUBROUTINE        S  O  L  V  E  B        SUTRA - VERSION 1284-2D W10.....


C     SUBROUTINE        S  O  L  V  E  B        SUTRA - VERSION 1284-2D W10.....
C                                                                       W20.....
C *** PURPOSE :                                                         W30.....
C ***  TO SOLVE THE MATRIX EQUATION BY:                                 W40.....
C ***    (1) DECOMPOSING THE MATRIX                                     W50.....
C ***    (2) MODIFYING THE RIGHT-HAND SIDE                              W60.....
C ***    (3) BACK-SUBSTITUTING FOR THE SOLUTION                         W70.....
C                                                                       W80.....
      SUBROUTINE SOLVEB(KKK,C,R,NNP,IHALFB,MAXNP,MAXBW)                 W90.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                               W100....
      DIMENSION C(MAXNP,MAXBW),R(MAXNP)                                 W110....
      IHBP=IHALFB+1                                                     W120....
C                                                                       W130....
C.....DECOMPOSE MATRIX C BY BANDED GAUSSIAN ELIMINATION FOR             W140....
C        NON-SYMMETRIC MATRIX                                           W150....
      IF(KKK-1) 5,5,50                                                  W160....
    5 NU=NNP-IHALFB                                                     W170....
      DO 20 NI=1,NU                                                     W180....
      PIVOTI=1.D0/C(NI,IHBP)                                            W190....
      NJ=NI+1                                                           W200....
      IB=IHBP                                                           W210....
      NK=NI+IHALFB                                                      W220....
      DO 10 NL=NJ,NK                                                    W230....
      IB=IB-1                                                           W240....
      A=-C(NL,IB)*PIVOTI                                                W250....
      C(NL,IB)=A                                                        W260....
      JB=IB+1                                                           W270....
      KB=IB+IHALFB                                                      W280....
      LB=IHBP-IB                                                        W290....
      DO10 MB=JB,KB                                                     W300....
      NB=LB+MB                                                          W310....
   10 C(NL,MB)=C(NL,MB)+A*C(NI,NB)                                      W320....
   20 CONTINUE                                                          W330....
      NR=NJ+1                                                           W340....
      NU=NNP-1                                                          W350....
      NK=NNP                                                            W360....
      DO 40 NI=NR,NU                                                    W370....
      PIVOTI=1.D0/(C(NI,IHBP))                                          W380....
      NJ=NI+1                                                           W390....
      IB=IHBP                                                           W400....
      DO 30 NL=NJ,NK                                                    W410....
      IB=IB-1                                                           W420....
      A=-C(NL,IB)*PIVOTI                                                W430....
      C(NL,IB)=A                                                        W440....
      JB=IB+1                                                           W450....
      KB=IB+IHALFB                                                      W460....
      LB=IHBP-IB                                                        W470....
      DO 30 MB=JB,KB                                                    W480....
      NB=LB+MB                                                          W490....
   30 C(NL,MB)=C(NL,MB)+A*C(NI,NB)                                      W500....
   40 CONTINUE                                                          W510....
      IF(KKK-1) 50,44,50                                                W520....
   44 RETURN                                                            W530....
C                                                                       W540....
C.....UPDATE RIGHT-HAND SIDE VECTOR, R                                  W550....
   50 NU=NNP+1                                                          W560....
      IBAND=2*IHALFB+1                                                  W570....
      DO 70 NI=2,IHBP                                                   W580....
      IB=IHBP-NI+1                                                      W590....
      NJ=1                                                              W600....
```

```
C     SUBROUTINE        S  O  L  V  E  B        SUTRA - VERSION 1284-2D W10.....


      SUM=0.0D0                                                         W610....
      DO 60 JB=IB,IHALFB                                                W620....
      SUM=SUM+C(NI,JB)*R(NJ)                                            W630....
   60 NJ=NJ+1                                                           W640....
   70 R(NI)=R(NI)+SUM                                                   W650....
      IB=1                                                              W660....
      NL=IHBP+1                                                         W670....
      DO 90 NI=NL,NNP                                                   W680....
      NJ=NI-IHBP+1                                                      W690....
      SUM=0.D0                                                          W700....
      DO 80 JB=IB,IHALFB                                                W710....
      SUM=SUM+C(NI,JB)*R(NJ)                                            W720....
   80 NJ=NJ+1                                                           W730....
   90 R(NI)=R(NI)+SUM                                                   W740....
C                                                                       W750....
C.....BACK SOLVE                                                        W760....
      R(NNP)=R(NNP)/C(NNP,IHBP)                                         W770....
      DO 110 IB=2,IHBP                                                  W780....
      NI=NU-IB                                                          W790....
      NJ=NI                                                             W800....
      MB=IHALFB+IB                                                      W810....
      SUM=0.D0                                                          W820....
      DO 100 JB=NL,MB                                                   W830....
      NJ=NJ+1                                                           W840....
  100 SUM=SUM+C(NI,JB)*R(NJ)                                            W850....
  110 R(NI)=(R(NI)-SUM)/C(NI,IHBP)                                      W860....
      MB=IBAND                                                          W870....
      DO 130 IB=NL,NNP                                                  W880....
      NI=NJ-IB                                                          W890....
      NJ=NI                                                             W900....
      SUM=0.D0                                                          W910....
      DO 120 JB=NL,MB                                                   W920....
      NJ=NJ+1                                                           W930....
  120 SUM=SUM+C(NI,JB)*R(NJ)                                            W940....
  130 R(NI)=(R(NI)-SUM)/C(NI,IHBP)                                      W950....
C                                                                       W960....
C                                                                       W970....
      RETURN                                                            W980....
      END                                                               W990....
```

```
C     SUBROUTINE      B  U  D  G  E  T      SUTRA - VERSION 1284-2D X10.....

C     SUBROUTINE      B  U  D  G  E  T      SUTRA - VERSION 1284-2D X10.....
C                                                                  X20.....
C *** PURPOSE :                                                    X30.....
C ***  TO CALCULATE AND OUTPUT FLUID MASS AND SOLUTE MASS OR       X40.....
C ***  ENERGY BUDGETS.                                            X50.....
C                                                                  X60.....
      SUBROUTINE BUDGET(ML,IBCT,VOL,SW,DSWDP,RHO,SOP,QIN,PVEC,PM1,X70.....
     1   PBC,QPLITR,IPBC,IQSOP,POR,UVEC,UM1,UM2,UIN,QUIN,IQSOU,UBC,X80.....
     2   CS1,CS2,CS3,SL,SR)                                        X90.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                          X100....
      CHARACTER*10 ADSMOD                                          X110....
      COMMON/MODSOR/ ADSMOD                                        X120....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,       X130....
     1   NSOP,NSOU,NBCN                                            X140....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,   X150....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                   X160....
      COMMON/PARAMS/ COMPFL,COMPMA,DRWDU,CW,CS,RHOS,DECAY,SIGMAW,SIGMAS,X170....
     1   RHOW0,URHOW0,VISC0,PRODF1,PRODS1,PRODF0,PRODS0,CHI1,CHI2  X180....
      COMMON/CONTRL/ GNU,UP,DTMULT,DTMAX,ME,ISSFLO,ISSTRA,ITCYC,   X190....
     1   NPCYC,NUCYC,NPRINT,IREAD,ISTORE,NOUMAT,IUNSAT             X200....
      CHARACTER*13 UNAME(2)                                        X210....
      DIMENSION QIN(NN),UIN(NN),IQSOP(NSOP),QUIN(NN),IQSOU(NSOU)   X220....
      DIMENSION IPBC(NBCN),UBC(NBCN),QPLITR(NBCN),PBC(NBCN)        X230....
      DIMENSION POR(NN),VOL(NN),PVEC(NN),UVEC(NN),SW(NN),DSWDP(NN),X240....
     1   RHO(NN),SOP(NN),PM1(NN),UM1(NN),UM2(NN),                  X250....
     2   CS1(NN),CS2(NN),CS3(NN),SL(NN),SR(NN)                     X260....
      DATA UNAME(1)/'CONCENTRATION'/,UNAME(2)/' TEMPERATURE '/     X270....
C                                                                  X280....
C                                                                  X290....
      MN=2                                                         X300....
      IF(IUNSAT.NE.0) IUNSAT=1                                     X310....
      IF(ME.EQ.-1) MN=1                                            X320....
      WRITE(6,10)                                                  X330....
   10 FORMAT(1H1)                                                  X340....
C.....SET UNSATURATED FLOW PARAMETERS, SW(I) AND DSWDP(I)          X350....
      IF(IUNSAT-1) 40,20,40                                        X360....
   20 DO 30 I=1,NN                                                 X370....
      IF(PVEC(I)) 25,27,27                                         X380....
   25 CALL UNSAT(SW(I),DSWDP(I),RELK,PVEC(I))                      X390....
      GOTO 30                                                      X400....
   27 SW(I)=1.0D0                                                  X410....
      DSWDP(I)=0.0D0                                               X420....
   30 CONTINUE                                                     X430....
C                                                                  X440....
C.....CALCULATE COMPONENTS OF FLUID MASS BUDGET                    X450....
   40 IF(ML-1) 50,50,1000                                          X460....
   50 CONTINUE                                                     X470....
      STPTOT=0.D0                                                  X480....
      STUTOT=0.D0                                                  X490....
      QINTOT=0.D0                                                  X500....
      DO 100 I=1,NN                                                X510....
      STPTOT=STPTOT+(1-ISSFLO/2)*RHO(I)*VOL(I)*                    X520....
     1   (SW(I)*SOP(I)+POR(I)*DSWDP(I))*(PVEC(I)-PM1(I))/DELTP     X530....
      STUTOT=STUTOT+(1-ISSFLO/2)*POR(I)*SW(I)*DRWDU*VOL(I)*        X540....
     1   (UM1(I)-UM2(I))/DLTUM1                                    X550....
      QINTOT=QINTOT+QIN(I)                                         X560....
  100 CONTINUE                                                     X570....
C                                                                  X580....
      QPLTOT=0.D0                                                  X590....
      DO 200 IP=1,NPBC                                             X600....
```

```
C     SUBROUTINE        B  U  D  G  E  T        SUTRA - VERSION 1284-2D X10.....


      I=IABS(IPBC(IP))                                                  X610....
      QPLITR(IP)=GNU*(PBC(IP)-PVEC(I))                                  X620....
      QPLTOT=QPLTOT+QPLITR(IP)                                          X630....
  200 CONTINUE                                                          X640....
C                                                                       X650....
C.....OUTPUT FLUID MASS BUDGET                                          X660....
      WRITE(6,300) IT,STPTOT,STUTOT,UNAME(MN),QINTOT,QPLTOT             X670....
  300 FORMAT(//11X,'F L U I D   M A S S   B U D G E T      AFTER TIME', X680....
     1   ' STEP ',I5,',     IN (MASS/SECOND)'///11X,1PD15.7,5X,         X690....
     2   'RATE OF CHANGE IN TOTAL STORED FLUID DUE TO PRESSURE CHANGE', X700....
     3   ', INCREASE(+)/DECREASE(-)',/11X,1PD15.7,5X,                   X710....
     2   'RATE OF CHANGE IN TOTAL STORED FLUID DUE TO ',A13,' CHANGE',  X720....
     3   ', INCREASE(+)/DECREASE(-)',                                   X730....
     3   /11X,1PD15.7,5X,'TOTAL OF FLUID SOURCES AND SINKS, ',          X740....
     4   'NET INFLOW(+)/NET OUTFLOW(-)'/11X,1PD15.7,5X,                 X750....
     5   'TOTAL OF FLUID FLOWS AT POINTS OF SPECIFIED PRESSURE, ',      X760....
     6   'NET INFLOW(+)/NET OUTFLOW(-)')                                X770....
C                                                                       X780....
      IF(IBCT.EQ.4) GOTO 600                                            X790....
      NSOPI=NSOP-1                                                      X800....
      INEGCT=0                                                          X810....
      DO 500 IQP=1,NSOPI                                                X820....
      I=IQSOP(IQP)                                                      X830....
      IF(I) 325,500,500                                                 X840....
  325 INEGCT=INEGCT+1                                                   X850....
      IF(INEGCT.EQ.1) WRITE(6,350)                                      X860....
  350 FORMAT(///22X,'TIME-DEPENDENT FLUID SOURCES OR SINKS'//22X,       X870....
     1   ' NODE',5X,'INFLOW(+)/OUTFLOW(-)'/37X,'  (MASS/SECOND)'//)     X880....
      WRITE(6,450) -I,QIN(-I)                                           X890....
  450 FORMAT(22X,I5,10X,1PD15.7)                                        X900....
  500 CONTINUE                                                          X910....
C                                                                       X920....
  600 IF(NPBC.EQ.0) GOTO 800                                            X930....
      WRITE(6,650)                                                      X940....
  650 FORMAT(///22X,'FLUID SOURCES OR SINKS DUE TO SPECIFIED PRESSURES',X950....
     1   //22X,' NODE',5X,'INFLOW(+)/OUTFLOW(-)'/37X,'  (MASS/SECOND)'/)X960....
      DO 700 IP=1,NPBC                                                  X970....
      I=IABS(IPBC(IP))                                                  X980....
      WRITE(6,450) I,QPLITR(IP)                                         X990....
  700 CONTINUE                                                          X1000...
C                                                                       X1010...
C.....CALCULATE COMPONENTS OF ENERGY OR SOLUTE MASS BUDGET              X1020...
  800 IF(ML-1) 1000,4500,1000                                           X1030...
 1000 CONTINUE                                                          X1040...
      FLDTOT=0.D0                                                       X1050...
      SLDTOT=0.D0                                                       X1060...
      P1FTOT=0.D0                                                       X1070...
      P1STOT=0.D0                                                       X1080...
      POFTOT=0.D0                                                       X1090...
      POSTOT=0.D0                                                       X1100...
      QQUTOT=0.D0                                                       X1110...
      QIUTOT=0.D0                                                       X1120...
C.....SET ADSORPTION PARAMETERS                                         X1130...
      IF(ME.EQ.-1.AND.ADSMOD.NE.'NONE      ')                           X1140...
     1   CALL ADSORB(CS1,CS2,CS3,SL,SR,UVEC)                            X1150...
      DO 1300 I=1,NN                                                    X1160...
      ESRV=POR(I)*SW(I)*RHO(I)*VOL(I)                                   X1170...
      EPRSV=(1.D0-POR(I))*RHOS*VOL(I)                                   X1180...
      DUDT=(1-ISSTRA)*(UVEC(I)-UM1(I))/DELTU                            X1190...
      FLDTOT=FLDTOT+ESRV*CW*DUDT                                        X1200...
```

```
      SLDTOT=SLDTOT+EPRSV*CS1(I)*DUDT                                   X1210...
      P1FTOT=P1FTOT+ESRV*PRODF1                                         X1220...
      P1STOT=P1STOT+EPRSV*PRODS1*(SL(I)*UVEC(I)+SR(I))                  X1230...
      POFTOT=POFTOT+ESRV*PRODF0                                         X1240...
      POSTOT=POSTOT+EPRSV*PRODS0                                        X1250...
      QQUTOT=QQUTOT+QUIN(I)                                             X1260...
      IF(QIN(I)) 1200,1200,1250                                         X1270...
 1200 QIUTOT=QIUTOT+QIN(I)*CW*UVEC(I)                                   X1280...
      GOTO 1300                                                         X1290...
 1250 QIUTOT=QIUTOT+QIN(I)*CW*UIN(I)                                    X1300...
 1300 CONTINUE                                                          X1310...
C                                                                       X1320...
      QPUTOT=0.D0                                                       X1330...
      DO 1500 IP=1,NPBC                                                 X1340...
      IF(QPLITR(IP)) 1400,1400,1450                                     X1350...
 1400 I=IABS(IPBC(IP))                                                  X1360...
      QPUTOT=QPUTOT+QPLITR(IP)*CW*UVEC(I)                               X1370...
      GOTO 1500                                                         X1380...
 1450 QPUTOT=QPUTOT+QPLITR(IP)*CW*UBC(IP)                               X1390...
 1500 CONTINUE                                                          X1400...
C                                                                       X1410...
      IF(ME) 1550,1550,1615                                             X1420...
C                                                                       X1430...
C.....OUTPUT SOLUTE MASS BUDGET                                         X1440...
 1550 WRITE(6,1600) IT,FLDTOT,SLDTOT,P1FTOT,P1STOT,POFTOT,POSTOT,       X1450...
     1   QIUTOT,QPUTOT,QQUTOT                                           X1460...
 1600 FORMAT(//11X,'S O L U T E   B U D G E T      AFTER TIME STEP ',I5,X1470...
     1   ',   IN (SOLUTE MASS/SECOND)'///11X,1PD15.7,5X,'NET RATE OF ', X1480...
     2   'INCREASE(+)/DECREASE(-) OF SOLUTE'/11X,1PD15.7,5X,           X1490...
     3   'NET RATE OF INCREASE(+)/DECREASE(-) OF ADSORBATE'/11X,1PD15.7,X1500...
     4   5X,'NET FIRST-ORDER PRODUCTION(+)/DECAY(-) OF SOLUTE'/11X,     X1510...
     5   1PD15.7,5X,'NET FIRST-ORDER PRODUCTION(+)/DECAY(-) OF ',       X1520...
     6   'ADSORBATE'/11X,1PD15.7,5X,'NET ZERO-ORDER PRODUCTION(+)/',    X1530...
     7   'DECAY(-) OF SOLUTE'/11X,1PD15.7,5X,'NET ZERO-ORDER ',         X1540...
     8   'PRODUCTION(+)/DECAY(-) OF ADSORBATE'/11X,1PD15.7,5X,          X1550...
     9   'NET GAIN(+)/LOSS(-) OF SOLUTE THROUGH FLUID SOURCES AND SINKS'X1560...
     *   /11X,1PD15.7,5X,'NET GAIN(+)/LOSS(-) OF SOLUTE THROUGH ',      X1570...
     1   'INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED PRESSURE'          X1580...
     2   /11X,1PD15.7,5X,'NET GAIN(+)/LOSS(-) OF SOLUTE THROUGH ',      X1590...
     3   'SOLUTE SOURCES AND SINKS')                                    X1600...
      GOTO 1645                                                         X1610...
C                                                                       X1620...
C.....OUTPUT ENERGY BUDGET                                              X1630...
 1615 WRITE(6,1635) IT,FLDTOT,SLDTOT,POFTOT,POSTOT,QIUTOT,QPUTOT,QQUTOT X1640...
 1635 FORMAT(//11X,'E N E R G Y   B U D G E T      AFTER TIME STEP ',I5,X1650...
     1   ',   IN (ENERGY/SECOND)'///11X,1PD15.7,5X,'NET RATE OF ',      X1660...
     2   'INCREASE(+)/DECREASE(-) OF ENERGY IN FLUID'/11X,1PD15.7,5X,   X1670...
     3   'NET RATE OF INCREASE(+)/DECREASE(-) OF ENERGY IN SOLID GRAINS'X1680...
     4   /11X,1PD15.7,5X,'NET ZERO-ORDER PRODUCTION(+)/LOSS(-) OF ',    X1690...
     5   'ENERGY IN FLUID'/11X,1PD15.7,5X,'NET ZERO-ORDER ',            X1700...
     6   'PRODUCTION(+)/LOSS(-) OF ENERGY IN SOLID GRAINS'              X1710...
     7   /11X,1PD15.7,5X,'NET GAIN(+)/LOSS(-) OF ENERGY THROUGH FLUID ',X1720...
     8   'SOURCES AND SINKS'/11X,1PD15.7,5X,'NET GAIN(+)/LOSS(-) OF ',  X1730...
     9   'ENERGY THROUGH INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED ',  X1740...
     *   'PRESSURE'/11X,1PD15.7,5X,'NET GAIN(+)/LOSS(-) OF ENERGY ',    X1750...
     1   'THROUGH ENERGY SOURCES AND SINKS')                            X1760...
C                                                                       X1770...
 1645 NSOPI=NSOP-1                                                      X1780...
      IF(NSOPI.EQ.0) GOTO 2000                                          X1790...
      IF(ME) 1649,1649,1659                                             X1800...
```

```
 1649 WRITE(6,1650)                                                     X1810...
 1650 FORMAT(///22X,'SOLUTE SOURCES OR SINKS AT FLUID SOURCES AND ',    X1820...
     1   'SINKS'//22X,' NODE',8X,'SOURCE(+)/SINK(-)'/32X,               X1830...
     2   '(SOLUTE MASS/SECOND)'/)                                       X1840...
      GOTO 1680                                                         X1850...
 1659 WRITE(6,1660)                                                     X1860...
 1660 FORMAT(///22X,'ENERGY SOURCES OR SINKS AT FLUID SOURCES AND ',    X1870...
     1   'SINKS'//22X,' NODE',8X,'SOURCE(+)/SINK(-)'/37X,               X1880...
     2   '(ENERGY/SECOND)'/)                                            X1890...
 1680 DO 1900 IQP=1,NSOPI                                               X1900...
      I=IABS(IQSOP(IQP))                                                X1910...
      IF(QIN(I)) 1700,1700,1750                                         X1920...
 1700 QU=QIN(I)*CW*UVEC(I)                                              X1930...
      GOTO 1800                                                         X1940...
 1750 QU=QIN(I)*CW*UIN(I)                                               X1950...
 1800 WRITE(6,450) I,QU                                                 X1960...
 1900 CONTINUE                                                          X1970...
C                                                                       X1980...
 2000 IF(NPBC.EQ.0) GOTO 4500                                           X1990...
      IF(ME) 2090,2090,2150                                             X2000...
 2090 WRITE(6,2100)                                                     X2010...
 2100 FORMAT(///22X,'SOLUTE SOURCES OR SINKS DUE TO FLUID INFLOWS OR ', X2020...
     1   'OUTFLOWS AT POINTS OF SPECIFIED PRESSURE'//22X,' NODE',8X,    X2030...
     2   'SOURCE(+)/SINK(-)'/32X,'(SOLUTE MASS/SECOND)'/)               X2040...
      GOTO 2190                                                         X2050...
 2150 WRITE(6,2160)                                                     X2060...
 2160 FORMAT(///22X,'ENERGY SOURCES OR SINKS DUE TO FLUID INFLOWS OR ', X2070...
     1   'OUTFLOWS AT POINTS OF SPECIFIED PRESSURE'//22X,' NODE',8X,    X2080...
     2   'SOURCE(+)/SINK(-)'/37X,'(ENERGY/SECOND)'/)                    X2090...
 2190 DO 2400 IP=1,NPBC                                                 X2100...
      I=IABS(IPBC(IP))                                                  X2110...
      IF(QPLITR(IP)) 2200,2200,2250                                     X2120...
 2200 QPU=QPLITR(IP)*CW*UVEC(I)                                         X2130...
      GOTO 2300                                                         X2140...
 2250 QPU=QPLITR(IP)*CW*UBC(IP)                                         X2150...
 2300 WRITE(6,450) I,QPU                                                X2160...
 2400 CONTINUE                                                          X2170...
C                                                                       X2180...
      IF(IBCT.EQ.4) GOTO 4500                                           X2190...
      NSOUI=NSOU-1                                                      X2200...
      INEGCT=0                                                          X2210...
      DO 3500 IQU=1,NSOUI                                               X2220...
      I=IQSOU(IQU)                                                      X2230...
      IF(I) 3400,3500,3500                                              X2240...
 3400 INEGCT=INEGCT+1                                                   X2250...
      IF(ME) 3450,3450,3460                                             X2260...
 3450 IF(INEGCT.EQ.1) WRITE(6,3455)                                     X2270...
 3455 FORMAT(///22X,'TIME-DEPENDENT SOLUTE SOURCES AND SINKS'//22X,     X2280...
     1   ' NODE',10X,'GAIN(+)/LOSS(-)'/30X,'  (SOLUTE MASS/SECOND)'//)  X2290...
      GOTO 3475                                                         X2300...
 3460 IF(INEGCT.EQ.1) WRITE(6,3465)                                     X2310...
 3465 FORMAT(///22X,'TIME-DEPENDENT ENERGY SOURCES AND SINKS'//22X,     X2320...
     1   ' NODE',10X,'GAIN(+)/LOSS(-)'/35X,'  (ENERGY/SECOND)'//)      X2330...
 3475 CONTINUE                                                          X2340...
      WRITE(6,3490) -I,QUIN(-I)                                         X2350...
 3490 FORMAT(22X,I5,10X,1PD15.7)                                        X2360...
 3500 CONTINUE                                                          X2370...
C                                                                       X2380...
C                                                                       X2390...
 4500 CONTINUE                                                          X2400...
```

```
C     SUBROUTINE        B  U  D  G  E  T        SUTRA - VERSION 1234-2D X10.....

C -                                                                     X2410...
      RETURN                                                            X2420...
      END                                                               X2430...
```

```
C     SUBROUTINE        S  T  O  R  E           SUTRA - VERSION 1284-2D Y10.....
```


```
C     SUBROUTINE        S  T  O  R  E           SUTRA - VERSION 1284-2D Y10.....
```


```
C                                                                         Y20.....
C *** PURPOSE :                                                           Y30.....
C ***  TO STORE RESULTS THAT MAY LATER BE USED TO RE-START                 Y40.....
C ***  THE SIMULATION.                                                    Y50.....
C                                                                         Y60.....
      SUBROUTINE STORE(PVEC,UVEC,PM1,UM1,CS1,RCIT,SW,PBC)                 Y70.....
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)                                 Y80.....
      COMMON/DIMS/ NN,NE,NIN,NBI,NB,NBHALF,NPINCH,NPBC,NUBC,              Y90.....
     1   NSOP,NSOU,NBCN                                                   Y100....
      COMMON/TIME/ DELT,TSEC,TMIN,THOUR,TDAY,TWEEK,TMONTH,TYEAR,          Y110....
     1   TMAX,DELTP,DELTU,DLTPM1,DLTUM1,IT,ITMAX                          Y120....
      DIMENSION PVEC(NN),UVEC(NN),PM1(NN),UM1(NN),CS1(NN),RCIT(NN),       Y130....
     1   SW(NN),PBC(NBCN)                                                 Y140....
C                                                                         Y150....
C.....REWIND UNIT-66 FOR WRITING RESULTS OF CURRENT TIME STEP             Y160....
      REWIND(66)                                                          Y170....
C                                                                         Y180....
C.....STORE TIME INFORMATION                                              Y190....
      WRITE(66,100) TSEC,DELTP,DELTU                                      Y200....
  100 FORMAT(4D20.10)                                                     Y210....
C                                                                         Y220....
C.....STORE SOLUTION                                                      Y230....
      WRITE(66,110) (PVEC(I),I=1,NN)                                      Y240....
      WRITE(66,110) (UVEC(I),I=1,NN)                                      Y250....
      WRITE(66,110) (PM1(I),I=1,NN)                                       Y260....
      WRITE(66,110) (UM1(I),I=1,NN)                                       Y270....
      WRITE(66,110) (CS1(I),I=1,NN)                                       Y280....
      WRITE(66,110) (RCIT(I),I=1,NN)                                      Y290....
      WRITE(66,110) (SW(I),I=1,NN)                                        Y300....
      WRITE(66,110) (PBC(IP),IP=1,NBCN)                                   Y310....
  110 FORMAT(4(1PD20.13))                                                 Y320....
C                                                                         Y330....
      ENDFILE(66)                                                         Y340....
C                                                                         Y350....
      RETURN                                                              Y360....
      END                                                                 Y370....
```


(p. 381 follows)

Appendix C:

Data File Listing for

Radial Energy Transport

Example

(p. 383 follows)

UNIT-5

```
SUTRA ENERGY TRANSPORT                                          #1 INPUT DATA HEADING
......... STEADY RADIAL FLOW WITH ENERGY TRANSPORT  -  SOLUTION CHECK .........
+++++++++ EXAMPLE RUN FOR SUTRA DOCUMENTATION - SECTION 6.3, PAGE 186 +++++++++
  132    65     7     0     2     2     2    00     4   222          #3 CONTROL NUMBERS
00000   +01   000    +1    +1                                          #4 MODE OPTIONS
    0.0000       1.0000+02                                         #5 NUMERICAL CONTROL
  225          4.021+03          1.296+25      99999         1.0     1.296+25  999   01
  225     0     0     0    00    00     1    01                       #7 OUTPUT OPTIONS
          01                                                       #8 ITERATION CONTROLS
    0.0000   4182.000     0.6000  1000.000     0.0000     0.0000     1.000  #9 FLUID
    0.0000    840.000     3.5000  2650.000                              #10 SOLID
NONE                                                              #11 ADSORPTION DATA
    0.0000     0.0000     0.0000     0.0000                           #12 PRODUCTION
    0.0000    -9.8000                                                 #13 GRAVITY
NODE      001.00     010.00     1.0000     0.2000                  #14A NODEWISE SCALES
    1     0.0000     0.0000      0.000      1.000                   #14B NODEWISE DATA
    2     0.0000     1.0000      0.000      1.000
    3     2.5000     0.0000     15.708      1.000
    4     2.5000     1.0000     15.708      1.000
    5     5.1520     0.0000     32.371      1.000
    6     5.1520     1.0000     32.371      1.000
    7     7.9654     0.0000     50.048      1.000
    8     7.9654     1.0000     50.048      1.000
    9    10.9498     0.0000     68.800      1.000
   10    10.9498     1.0000     68.800      1.000
   11    14.1158     0.0000     88.692      1.000
   12    14.1158     1.0000     88.692      1.000
   13    17.4743     0.0000    109.794      1.000
   14    17.4743     1.0000    109.794      1.000
   15    21.0370     0.0000    132.179      1.000
   16    21.0370     1.0000    132.179      1.000
   17    24.8164     0.0000    155.926      1.000
   18    24.8164     1.0000    155.926      1.000
   19    28.8257     0.0000    181.117      1.000
   20    28.8257     1.0000    181.117      1.000
   21    33.0789     0.0000    207.840      1.000
   22    33.0789     1.0000    207.840      1.000
   23    37.5907     0.0000    236.189      1.000
   24    37.5907     1.0000    236.189      1.000
   25    42.3768     0.0000    266.261      1.000
   26    42.3768     1.0000    266.261      1.000
   27    47.4541     0.0000    298.163      1.000
   28    47.4541     1.0000    298.163      1.000
   29    52.8402     0.0000    332.004      1.000
   30    52.8402     1.0000    332.004      1.000
   31    58.5538     0.0000    367.904      1.000
   32    58.5538     1.0000    367.904      1.000
   33    64.6150     0.0000    405.987      1.000
   34    64.6150     1.0000    405.987      1.000
   35    71.0447     0.0000    446.387      1.000
   36    71.0447     1.0000    446.387      1.000
   37    77.8655     0.0000    489.243      1.000
   38    77.8655     1.0000    489.243      1.000
   39    85.1012     0.0000    534.706      1.000
   40    85.1012     1.0000    534.706      1.000
   41    92.7769     0.0000    582.934      1.000
   42    92.7769     1.0000    582.934      1.000
   43   100.9194     0.0000    634.095      1.000
```

44	100.9194	1.0000	634.095	1.000
45	109.5571	0.0000	688.367	1.000
46	109.5571	1.0000	688.367	1.000
47	118.7202	0.0000	745.940	1.000
48	118.7202	1.0000	745.940	1.000
49	128.4405	0.0000	807.015	1.000
50	128.4405	1.0000	807.015	1.000
51	138.7520	0.0000	871.804	1.000
52	138.7520	1.0000	871.804	1.000
53	149.6907	0.0000	940.533	1.000
54	149.6907	1.0000	940.533	1.000
55	161.2946	0.0000	1013.443	1.000
56	161.2946	1.0000	1013.443	1.000
57	173.6042	0.0000	1090.786	1.000
58	173.6042	1.0000	1090.786	1.000
59	186.6625	0.0000	1172.834	1.000
60	186.6625	1.0000	1172.834	1.000
61	200.5150	0.0000	1259.872	1.000
62	200.5150	1.0000	1259.872	1.000
63	215.2099	0.0000	1352.203	1.000
64	215.2099	1.0000	1352.203	1.000
65	230.7986	0.0000	1450.149	1.000
66	230.7986	1.0000	1450.149	1.000
67	247.3354	0.0000	1554.052	1.000
68	247.3354	1.0000	1554.052	1.000
69	264.8778	0.0000	1664.275	1.000
70	264.8778	1.0000	1664.275	1.000
71	283.4872	0.0000	1781.201	1.000
72	283.4872	1.0000	1781.201	1.000
73	303.2283	0.0000	1905.238	1.000
74	303.2283	1.0000	1905.238	1.000
75	324.1701	0.0000	2036.819	1.000
76	324.1701	1.0000	2036.819	1.000
77	346.3856	0.0000	2176.403	1.000
78	346.3856	1.0000	2176.403	1.000
79	369.9521	0.0000	2324.476	1.000
80	369.9521	1.0000	2324.476	1.000
81	394.9519	0.0000	2481.554	1.000
82	394.9519	1.0000	2481.554	1.000
83	419.1538	0.0000	2633.619	1.000
84	419.1538	1.0000	2633.619	1.000
85	443.3557	0.0000	2785.684	1.000
86	443.3557	1.0000	2785.684	1.000
87	467.5576	0.0000	2937.749	1.000
88	467.5576	1.0000	2937.749	1.000
89	491.7595	0.0000	3089.813	1.000
90	491.7595	1.0000	3089.813	1.000
91	515.9614	0.0000	3241.878	1.000
92	515.9614	1.0000	3241.878	1.000
93	540.1633	0.0000	3393.943	1.000
94	540.1633	1.0000	3393.943	1.000
95	564.3652	0.0000	3546.008	1.000
96	564.3652	1.0000	3546.008	1.000
97	588.5671	0.0000	3698.073	1.000
98	588.5671	1.0000	3698.073	1.000
99	612.7690	0.0000	3850.138	1.000
100	612.7690	1.0000	3850.138	1.000
101	636.9709	0.0000	4002.203	1.000

102	636.9709	1.0000	4002.203	1.000
103	661.1730	0.0000	4154.269	1.000
104	661.1730	1.0000	4154.269	1.000
105	685.3749	0.0000	4306.333	1.000
106	685.3749	1.0000	4306.333	1.000
107	709.5768	0.0000	4458.398	1.000
108	709.5768	1.0000	4458.398	1.000
109	733.7787	0.0000	4610.463	1.000
110	733.7787	1.0000	4610.463	1.000
111	757.9806	0.0000	4762.528	1.000
112	757.9806	1.0000	4762.528	1.000
113	782.1825	0.0000	4914.593	1.000
114	782.1825	1.0000	4914.593	1.000
115	806.3844	0.0000	5066.658	1.000
116	806.3844	1.0000	5066.658	1.000
117	830.5863	0.0000	5218.723	1.000
118	830.5863	1.0000	5218.723	1.000
119	854.7882	0.0000	5370.788	1.000
120	854.7882	1.0000	5370.788	1.000
121	878.9901	0.0000	5522.853	1.000
122	878.9901	1.0000	5522.853	1.000
123	903.1920	0.0000	5674.918	1.000
124	903.1920	1.0000	5674.918	1.000
125	927.3940	0.0000	5826.983	1.000
126	927.3940	1.0000	5826.983	1.000
127	951.5959	0.0000	5979.049	1.000
128	951.5959	1.0000	5979.049	1.000
129	975.7979	0.0000	6131.113	1.000
130	975.7979	1.0000	6131.113	1.000
131	999.9998	0.0000	6283.178	1.000
132	999.9998	1.0000	6283.178	1.000

ELEMENT	1.02-11	1.02-11	0.000	10.0	10.0	0.0	#15A SCAL
1	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	#15B
2	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	ELEMENT-
3	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	WISE
4	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	DATA
5	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
6	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
7	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
8	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
9	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
10	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
11	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
12	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
13	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
14	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
15	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
16	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
17	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
18	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
19	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
20	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
21	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
22	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
23	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
24	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
25	1.00E+00	1.00E+00	1.00E+00	1.0	1.00	1.000	
26	1.00E+00	1.00E+00	1.00E+00.	1.0	1.00	1.000	

```
        27  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        28  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        29  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        30  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        31  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        32  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        33  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        34  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        35  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        36  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        37  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        38  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        39  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        40  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        41  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        42  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        43  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        44  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        45  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        46  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        47  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        48  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        49  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        50  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        51  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        52  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        53  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        54  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        55  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        56  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        57  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        58  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        59  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        60  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        61  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        62  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        63  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        64  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
        65  1.00E+00  1.00E+00  1.00E+00       1.0      1.00      1.000
         1       156.25D-0         1.00D-0                          #17 FLUID
         2       156.25D-0         1.00D-0                            SOURCES
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
  132           0.00000000                                          #19 SPECIFIED
  131       98000.00000000                                            PRESSURES
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
    1           1.00000000                                          #20 SPECIFIED
    2           1.00000000                                            TEMPERATURES
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
       45                                                         #21 OBSERVATION NODES
   34   52   64   72   00
     1    1    3    4    2                                       #22 NODAL INCIDENCES
     2    3    5    6    4
     3    5    7    8    6
     4    7    9   10    8
     5    9   11   12   10
     6   11   13   14   12
     7   13   15   16   14
     8   15   17   18   16
```

9	17	19	20	18
10	19	21	22	20
11	21	23	24	22
12	23	25	26	24
13	25	27	28	26
14	27	29	30	28
15	29	31	32	30
16	31	33	34	32
17	33	35	36	34
18	35	37	38	36
19	37	39	40	38
20	39	41	42	40
21	41	43	44	42
22	43	45	46	44
23	45	47	48	46
24	47	49	50	48
25	49	51	52	50
26	51	53	54	52
27	53	55	56	54
28	55	57	58	56
29	57	59	60	58
30	59	61	62	60
31	61	63	64	62
32	63	65	66	64
33	65	67	68	66
34	67	69	70	68
35	69	71	72	70
36	71	73	74	72
37	73	75	76	74
38	75	77	78	76
39	77	79	80	78
40	79	81	82	80
41	81	83	84	82
42	83	85	86	84
43	85	87	88	86
44	87	89	90	88
45	89	91	92	90
46	91	93	94	92
47	93	95	96	94
48	95	97	98	96
49	97	99	100	98
50	99	101	102	100
51	101	103	104	102
52	103	105	106	104
53	105	107	108	106
54	107	109	110	108
55	109	111	112	110
56	111	113	114	112
57	113	115	116	114
58	115	117	118	116
59	117	119	120	118
60	119	121	122	120
61	121	123	124	122
62	123	125	126	124
63	125	127	128	126
64	127	129	130	128
65	129	131	132	130

UNIT-55

0.00000000000D+00

0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+06
0D+06	0D+06	0D+06	0D+05
0D+06	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+05
0D+05	0D+05	0D+05	0D+04
0D+05	0D+04	0D+05	0D+04
0D+05	0D+04	0D+04	0D+00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00
0.00D-00	0.00D-00	0.00D-00	0.00D-00

```
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
0.00D-00          0.00D-00          0.00D-00          0.00D-00
```

(p. 391 follows)

Appendix D:

Output Listing for

Radial Energy Transport

Example

(p. 393 follows)

****

****

****

****

```
 SSSS   UU  UU  TTTTTT  RRRRR     AA
SS   S  UU  UU  T TT T  RR  RR   AAAA
SSSS    UU  UU    TT    RRRRR   AA  AA
    SS  UU  UU    TT    RR R    AAAAAA
SS  SS  UU  UU    TT    RR RR   AA  AA
 SSSS    UUUU     TT    RR  RR  AA  AA
```

U N I T E D S T A T E S G E O L O G I C A L S U R V E Y

SUBSURFACE FLOW AND TRANSPORT SIMULATION MODEL

-VERSION 1284-20-

* SATURATED-UNSATURATED FLOW AND SOLUTE OR ENERGY TRANSPORT *

****

****

****

****

****

***** S U T R A E N E R G Y T R A N S P O R T S I M U L A T I O N *****

****

..... STEADY RADIAL FLOW WITH ENERGY TRANSPORT - SOLUTION CHECK .....

+++++++++ EXAMPLE RUN FOR SUTRA DOCUMENTATION - SECTION 6.3, PAGE 186 +++++++++

S I M U L A T I O N M O D E O P T I O N S

- ASSUME SATURATED FLOW ONLY
- ASSUME STEADY-STATE FLOW FIELD CONSISTENT WITH INITIAL TEMPERATURE CONDITIONS
- ALLOW TIME-DEPENDENT TRANSPORT
- COLD START - BEGIN NEW SIMULATION
- STORE RESULTS AFTER EACH TIME STEP ON UNIT-66 AS BACK-UP AND FOR USE IN A SIMULATION RE-START

S I M U L A T I O N C O N T R O L N U M B E R S

132	NUMBER OF NODES IN FINITE-ELEMENT MESH
65	NUMBER OF ELEMENTS IN MESH
7	ESTIMATED MAXIMUM FULL BAND WIDTH FOR MESH
0	EXACT NUMBER OF PINCH NODES IN MESH
2	EXACT NUMBER OF NODES IN MESH AT WHICH PRESSURE IS A SPECIFIED CONSTANT OR FUNCTION OF TIME
2	EXACT NUMBER OF NODES IN MESH AT WHICH TEMPERATURE IS A SPECIFIED CONSTANT OR FUNCTION OF TIME
2	EXACT NUMBER OF NODES AT WHICH FLUID INFLOW OR OUTFLOW IS A SPECIFIED CONSTANT OR FUNCTION OF TIME
0	EXACT NUMBER OF NODES AT WHICH A SOURCE OR SINK OF ENERGY IS A SPECIFIED CONSTANT OR FUNCTION OF TIME
4	EXACT NUMBER OF NODES AT WHICH PRESSURE AND TEMPERATURE WILL BE OBSERVED
222	MAXIMUM NUMBER OF TIME STEPS ON WHICH OBSERVATIONS WILL BE MADE

N U M E R I C A L C O N T R O L D A T A

0.00000	"UPSTREAM WEIGHTING" FACTOR
1.0000D+02	SPECIFIED PRESSURE BOUNDARY CONDITION FACTOR

T E M P O R A L C O N T R O L A N D S O L U T I O N C Y C L I N G D A T A

225	MAXIMUM ALLOWED NUMBER OF TIME STEPS
4.02100+03	INITIAL TIME STEP (IN SECONDS)
1.29600+25	MAXIMUM ALLOWED SIMULATION TIME (IN SECONDS)
99999	TIME STEP MULTIPLIER CYCLE (IN TIME STEPS)
1.00000	MULTIPLICATION FACTOR FOR TIME STEP CHANGE
1.29600+25	MAXIMUM ALLOWED TIME STEP (IN SECONDS)
999	FLOW SOLUTION CYCLE (IN TIME STEPS)
1	TRANSPORT SOLUTION CYCLE (IN TIME STEPS)

O U T P U T C O N T R O L S A N D O P T I O N S

225 PRINTED OUTPUT CYCLE (IN TIME STEPS)

- CANCEL PRINT OF NODE COORDINATES, THICKNESSES AND POROSITIES
- CANCEL PRINT OF ELEMENT PERMEABILITIES AND DISPERSIVITIES
- CANCEL PRINT OF NODE AND PINCH NODE INCIDENCES IN EACH ELEMENT

- CANCEL PLOT OF PRESSURES
- CANCEL PLOT OF TEMPERATURES

- CALCULATE AND PRINT VELOCITIES AT ELEMENT CENTROIDS ON EACH TIME STEP WITH OUTPUT

- CALCULATE AND PRINT FLUID AND ENERGY BUDGETS ON EACH TIME STEP WITH OUTPUT

I T E R A T I O N C O N T R O L D A T A

NON-ITERATIVE SOLUTION

C O N S T A N T P R O P E R T I E S O F F L U I D A N D S O L I D M A T R I X

```
     0.00000D-01      COMPRESSIBILITY OF FLUID
     0.00000D-01      COMPRESSIBILITY OF POROUS MATRIX

     4.18200D+03      SPECIFIC HEAT CAPACITY OF FLUID
     8.40000D+02      SPECIFIC HEAT CAPACITY OF SOLID GRAIN

  FLUID VISCOSITY IS CALCULATED BY SUTRA AS A FUNCTION OF TEMPERATURE IN UNITS OF [kg/(m*s)]

     1.00000D+00      VISCO, CONVERSION FACTOR FOR VISCOSITY UNITS,  [desired units] = VISCO*[kg/(m*s)]

     2.65000D+03      DENSITY OF A SOLID GRAIN

  FLUID DENSITY, RHOW
  CALCULATED BY SUTRA IN TERMS OF TEMPERATURE, U, AS:
  RHOw = RHOW0 + DRWDU*(U-URHOW0)

     1.00000D+03      FLUID BASE DENSITY, RHOW0
     0.00000D-01      COEFFICIENT OF DENSITY CHANGE WITH TEMPERATURE, DRWDU
     0.00000D-01      TEMPERATURE, URHOW0, AT WHICH FLUID DENSITY IS AT BASE VALUE, RHOW0

     6.00000D-01      THERMAL CONDUCTIVITY OF FLUID
     3.50000D+00      THERMAL CONDUCTIVITY OF SOLID GRAIN
```

P R O D U C T I O N A N D L O S S O F E N E R G Y

```
  PRODUCTION RATE (+)
  LOSS RATE (-)

     0.00000D-01      ZERO-ORDER RATE OF ENERGY PRODUCTION/LOSS IN FLUID
     0.00000D-01      ZERO-ORDER RATE OF ENERGY PRODUCTION/LOSS IN SOLID GRAINS
```

C O O R D I N A T E O R I E N T A T I O N T O G R A V I T Y

```
  COMPONENT OF GRAVITY VECTOR
  IN +X DIRECTION, GRAVX
     0.00000D-01      GRAVX = -GRAV * D(ELEVATION)/DX

  COMPONENT OF GRAVITY VECTOR
  IN +Y DIRECTION, GRAVY
    -9.80000D+00      GRAVY = -GRAV * D(ELEVATION)/DY
```

N O D E I N F O R M A T I O N

PRINTOUT OF NODE COORDINATES, THICKNESSES AND POROSITIES CANCELLED.

SCALE FACTORS :

1.00000+00	X-SCALE
1.00000+01	Y-SCALE
1.00000+00	THICKNESS FACTOR
2.00000-01	POROSITY FACTOR

E L E M E N T I N F O R M A T I O N

PRINTOUT OF ELEMENT PERMEABILITIES AND DISPERSIVITIES CANCELLED.

SCALE FACTORS :

1.02000-11	MAXIMUM PERMEABILITY FACTOR
1.02000-11	MINIMUM PERMEABILITY FACTOR
0.00000-01	ANGLE FROM +X TO MAXIMUM DIRECTION FACTOR
1.00000+01	MAXIMUM LONGITUDINAL DISPERSIVITY FACTOR
1.00000+01	MINIMUM LONGITUDINAL DISPERSIVITY FACTOR
0.00000-01	TRANSVERSE DISPERSIVITY FACTOR

F L U I D S O U R C E D A T A

**** NODES AT WHICH FLUID INFLOWS OR OUTFLOWS ARE SPECIFIED ****

NODE NUMBER (MINUS INDICATES TIME-VARYING FLOW OR TEMPERATURE)	FLUID INFLOW(+)/OUTFLOW(-) (FLUID MASS/SECOND)	TEMPERATURE [DEGREES CELCIUS] OF INFLOWING FLUID
1	1.5625000E+02	1.0000000E+00
2	1.5625000E+02	1.0000000E+00

B O U N D A R Y C O N D I T I O N S

**** NODES AT WHICH PRESSURES ARE SPECIFIED ****

(AS WELL AS TEMPERATURE [DEGREES CELCIUS] OF ANY
FLUID INFLOW WHICH MAY OCCUR AT THE POINT
OF SPECIFIED PRESSURE)

NODE	PRESSURE	TEMPERATURE
132	0.00000000000000D-01	0.00000000000000D-01
131	9.80000000000000D+04	0.00000000000000D-01

**** NODES AT WHICH TEMPERATURES ARE SPECIFIED TO BE INDEPENDENT OF LOCAL FLOWS AND FLUID SOURCES ****

NODE	TEMPERATURE
1	1.00000000000000D+00
2	1.00000000000000D+00

O B S E R V A T I O N N O D E S

**** NODES AT WHICH OBSERVATIONS WILL BE MADE EVERY 45 TIME STEPS ****

34 52 64 72

M E S H C O N N E C T I O N D A T A

PRINTOUT OF NODAL INCIDENCES AND PINCH NODE CONNECTIONS CANCELLED.

**** MESH ANALYSIS ****

ACTUAL MAXIMUM BANDWIDTH, 7, WAS CALCULATED IN ELEMENT 1
-------INPUT BANDWIDTH IS 7

E N D O F I N P U T F R O M U N I T - 5

INITIAL CONDITIONS

TIME INCREMENT 4.02100+03 SECONDS

ELAPSED TIME :
0.00000-01 SECONDS
0.00000-01 MINUTES
0.00000-01 HOURS
0.00000-01 DAYS
0.00000-01 WEEKS
0.00000-01 MONTHS
0.00000-01 YEARS

TEMPERATURE

NODE		NODE		NODE		NODE		NODE		NODE	
1	0.000000000	2	0.000000000	3	0.000000000	4	0.000000000	5	0.000000000	6	0.000000000
7	0.000000000	8	0.000000000	9	0.000000000	10	0.000000000	11	0.000000000	12	0.000000000
13	0.000000000	14	0.000000000	15	0.000000000	16	0.000000000	17	0.000000000	18	0.000000000
19	0.000000000	20	0.000000000	21	0.000000000	22	0.000000000	23	0.000000000	24	0.000000000
25	0.000000000	26	0.000000000	27	0.000000000	28	0.000000000	29	0.000000000	30	0.000000000
31	0.000000000	32	0.000000000	33	0.000000000	34	0.000000000	35	0.000000000	36	0.000000000
37	0.000000000	38	0.000000000	39	0.000000000	40	0.000000000	41	0.000000000	42	0.000000000
43	0.000000000	44	0.000000000	45	0.000000000	46	0.000000000	47	0.000000000	48	0.000000000
49	0.000000000	50	0.000000000	51	0.000000000	52	0.000000000	53	0.000000000	54	0.000000000
55	0.000000000	56	0.000000000	57	0.000000000	58	0.000000000	59	0.000000000	60	0.000000000
61	0.000000000	62	0.000000000	63	0.000000000	64	0.000000000	65	0.000000000	66	0.000000000
67	0.000000000	68	0.000000000	69	0.000000000	70	0.000000000	71	0.000000000	72	0.000000000
73	0.000000000	74	0.000000000	75	0.000000000	76	0.000000000	77	0.000000000	78	0.000000000
79	0.000000000	80	0.000000000	81	0.000000000	82	0.000000000	83	0.000000000	84	0.000000000
85	0.000000000	86	0.000000000	87	0.000000000	88	0.000000000	89	0.000000000	90	0.000000000
91	0.000000000	92	0.000000000	93	0.000000000	94	0.000000000	95	0.000000000	96	0.000000000
97	0.000000000	98	0.000000000	99	0.000000000	100	0.000000000	101	0.000000000	102	0.000000000
103	0.000000000	104	0.000000000	105	0.000000000	106	0.000000000	107	0.000000000	108	0.000000000
109	0.000000000	110	0.000000000	111	0.000000000	112	0.000000000	113	0.000000000	114	0.000000000
115	0.000000000	116	0.000000000	117	0.000000000	118	0.000000000	119	0.000000000	120	0.000000000
121	0.000000000	122	0.000000000	123	0.000000000	124	0.000000000	125	0.000000000	126	0.000000000
127	0.000000000	128	0.000000000	129	0.000000000	130	0.000000000	131	0.000000000	132	0.000000000

RESULTS FOR TIME STEP 0

S T E A D Y - S T A T E P R E S S U R E

NODE		NODE		NODE		NODE		NODE		NODE	
1	6.90140418D+06	2	6.80340418D+06	3	5.18917084D+06	4	5.09117084D+06	5	4.59575119D+06	6	4.49775119D+06
7	4.22851288D+06	8	4.13051288D+06	9	3.95836038D+06	10	3.86036038D+06	11	3.74209040D+06	12	3.64409040D+06
13	3.56005355D+06	14	3.46205355D+06	15	3.40165318D+06	16	3.30365318D+06	17	3.26052430D+06	18	3.16252430D+06
19	3.13254873D+06	20	3.03454873D+06	21	3.01490791D+06	22	2.91690791D+06	23	2.90559226D+06	24	2.80759226D+06
25	2.80311376D+06	26	2.70511376D+06	27	2.70633698D+06	28	2.60833698D+06	29	2.61438468D+06	30	2.51638468D+06
31	2.52656078D+06	32	2.42856078D+06	33	2.44230055D+06	34	2.34430055D+06	35	2.36114759D+06	36	2.26314759D+06
37	2.28271889D+06	38	2.18471889D+06	39	2.20669571D+06	40	2.10869571D+06	41	2.13281010D+06	42	2.03481010D+06
43	2.06083195D+06	44	1.96283195D+06	45	1.99056376D+06	46	1.89256376D+06	47	1.92183411D+06	48	1.82383411D+06
49	1.85449542D+06	50	1.75649542D+06	51	1.78841668D+06	52	1.69041668D+06	53	1.72348291D+06	54	1.62548291D+06
55	1.65959354D+06	56	1.56159354D+06	57	1.59665817D+06	58	1.49865817D+06	59	1.53459601D+06	60	1.43659601D+06
61	1.47333525D+06	62	1.37533525D+06	63	1.41281164D+06	64	1.31481164D+06	65	1.35296620D+06	66	1.25496620D+06
67	1.29374648D+06	68	1.19574648D+06	69	1.23510531D+06	70	1.13710531D+06	71	1.17699850D+06	72	1.07899850D+06
73	1.11938714D+06	74	1.02138714D+06	75	1.06223468D+06	76	9.64234682D+05	77	1.00550824D+06	78	9.07508235D+05
79	9.49177997D+05	80	8.51177997D+05	81	8.93215912D+05	82	7.95215912D+05	83	8.42314132D+05	84	7.44314132D+05
85	7.94268948D+05	86	6.96268948D+05	87	7.48776773D+05	88	6.50776773D+05	89	7.05579970D+05	90	6.07579970D+05
91	6.64458031D+05	92	5.66458031D+05	93	6.25220775D+05	94	5.27220775D+05	95	5.87703015D+05	96	4.89703015D+05
97	5.51760373D+05	98	4.53760373D+05	99	5.17265920D+05	100	4.19265920D+05	101	4.84107476D+05	102	3.86107476D+05
103	4.52185149D+05	104	3.54185149D+05	105	4.21410577D+05	106	3.23410577D+05	107	3.91703857D+05	108	2.93703857D+05
109	3.62993370D+05	110	2.64993370D+05	111	3.35214467D+05	112	2.37214467D+05	113	3.08308592D+05	114	2.10308592D+05
115	2.82222542D+05	116	1.84222542D+05	117	2.56907834D+05	118	1.58907834D+05	119	2.32320164D+05	120	1.34320164D+05
121	2.08418934D+05	122	1.10418934D+05	123	1.85166859D+05	124	8.71668593D+04	125	1.62529516D+05	126	6.45295156D+04
127	1.40475416D+05	128	4.24754155D+04	129	1.18975085D+05	130	2.09750849D+04	131	9.80015625D+04	132	1.56250000D+00

F L U I D M A S S B U D G E T AFTER TIME STEP 0, IN (MASS/SECOND)

0.0000000D-01	RATE OF CHANGE IN TOTAL STORED FLUID DUE TO PRESSURE CHANGE, INCREASE(+)/DECREASE(-)
0.0000000D-01	RATE OF CHANGE IN TOTAL STORED FLUID DUE TO TEMPERATURE CHANGE, INCREASE(+)/DECREASE(-)
3.1250000D+02	TOTAL OF FLUID SOURCES AND SINKS, NET INFLOW(+)/NET OUTFLOW(-)
-3.1250000D+02	TOTAL OF FLUID FLOWS AT POINTS OF SPECIFIED PRESSURE, NET INFLOW(+)/NET OUTFLOW(-)

FLUID SOURCES OR SINKS DUE TO SPECIFIED PRESSURES

NODE	INFLOW(+)/OUTFLOW(-) (MASS/SECOND)
132	-1.5625000D+02
131	-1.5625000D+02

RESULTS FOR TIME STEP 1

TIME INCREMENT 4.0210D+03 SECONDS

ELAPSED TIME : 4.0210D+03 SECONDS
6.7017D+01 MINUTES
1.1169D+00 HOURS
4.6539D-02 DAYS
6.6485D-03 WEEKS
1.5290D-03 MONTHS
1.2742D-04 YEARS

T E M P E R A T U R E

NODE		NODE		NODE		NODE		NODE		NODE	
1	1.000000000	2	1.000000000	3	0.822324058	4	0.822324058	5	0.624022580	6	0.624022580
7	0.436159008	8	0.436159008	9	0.279412944	10	0.279412944	11	0.163103312	12	0.163103312
13	0.086203260	14	0.086203260	15	0.040972094	16	0.040972094	17	0.017390730	18	0.017390730
19	0.006545764	20	0.006545764	21	0.002169799	22	0.002169799	23	0.000629264	24	0.000629264
25	0.000158676	26	0.000158676	27	0.000034593	28	0.000034593	29	0.000006488	30	0.000006488
31	0.000001042	32	0.000001042	33	0.000000143	34	0.000000143	35	0.000000017	36	0.000000017
37	0.000000002	38	0.000000002	39	0.000000000	40	0.000000000	41	0.000000000	42	0.000000000
43	0.000000000	44	0.000000000	45	0.000000000	46	0.000000000	47	0.000000000	48	0.000000000
49	0.000000000	50	0.000000000	51	0.000000000	52	0.000000000	53	0.000000000	54	0.000000000
55	0.000000000	56	0.000000000	57	0.000000000	58	0.000000000	59	0.000000000	60	0.000000000
61	0.000000000	62	0.000000000	63	0.000000000	64	0.000000000	65	0.000000000	66	0.000000000
67	0.000000000	68	0.000000000	69	0.000000000	70	0.000000000	71	0.000000000	72	0.000000000
73	0.000000000	74	0.000000000	75	0.000000000	76	0.000000000	77	0.000000000	78	0.000000000
79	0.000000000	80	0.000000000	81	0.000000000	82	0.000000000	83	0.000000000	84	0.000000000
85	0.000000000	86	0.000000000	87	0.000000000	88	0.000000000	89	0.000000000	90	0.000000000
91	0.000000000	92	0.000000000	93	0.000000000	94	0.000000000	95	0.000000000	96	0.000000000
97	0.000000000	98	0.000000000	99	0.000000000	100	0.000000000	101	0.000000000	102	0.000000000
103	0.000000000	104	0.000000000	105	0.000000000	106	0.000000000	107	0.000000000	108	0.000000000
109	0.000000000	110	0.000000000	111	0.000000000	112	0.000000000	113	0.000000000	114	0.000000000
115	0.000000000	116	0.000000000	117	0.000000000	118	0.000000000	119	0.000000000	120	0.000000000
121	0.000000000	122	0.000000000	123	0.000000000	124	0.000000000	125	0.000000000	126	0.000000000
127	0.000000000	128	0.000000000	129	0.000000000	130	0.000000000	131	0.000000000	132	0.000000000

S T E A D Y - S T A T E F L U I D V E L O C I T Y

M A G N I T U D E AT CENTROID OF ELEMENT

ELEMENT		ELEMENT		ELEMENT		ELEMENT		ELEMENT		ELEMENT	
1	1.989432140-02	2	6.49971921D-03	3	3.791601450-03	4	2.629408990-03	5	1.984227770-03	6	1.57441835D-03
7	1.291466400-03	8	1.084673990-03	9	9.27181398D-04	10	8.034307130-04	11	7.037828610-04	12	6.21952433D-04
13	5.536617860-04	14	4.959002930-04	15	4.464872530-04	16	4.038036360-04	17	3.666231020-04	18	3.33999551D-04
19	3.051909810-04	20	2.79607029D-04	21	2.567728460-04	22	2.363016860-04	23	2.178752530-04	24	2.01229269D-04
25	1.861427590-04	26	1.72429300D-04	27	1.599303160-04	28	1.485104520-04	29	1.380532070-04	30	1.28457775D-04
31	1.196366870-04	32	1.11513472D-04	33	1.040210030-04	34	9.710013930-05	35	9.069864370-05	36	8.47701535D-05
37	7.927333370-05	38	7.41712637D-05	39	6.943088230-05	40	6.502248220-05	41	6.109275290-05	42	5.76642421D-05
43	5.460009750-05	44	5.18451739D-05	45	4.935490380-05	46	4.709289180-05	47	4.502913640-05	48	4.31386673D-05
49	4.140053850-05	50	3.97970491D-05	51	3.831313340-05	52	3.693590600-05	53	3.565426030-05	54	3.44585720D-05
55	3.334047800-05	56	3.22926623D-05	57	3.130870070-05	58	3.038292890-05	59	2.951033340-05	60	2.86864603D-05
61	2.790733980-05	62	2.71694218D-05	63	2.646952000-05	64	2.580477450-05	65	2.517260150-05		

S T E A D Y - S T A T E F L U I D V E L O C I T Y

A N G L E IN DEGREES FROM +X-AXIS TO FLOW DIRECTION AT CENTROID OF ELEMENT

ELEMENT		ELEMENT		ELEMENT		ELEMENT		ELEMENT		ELEMENT	
1	0.000000000D-01	2	0.000000000D-01	3	0.000000000D-01	4	0.000000000D-01	5	0.000000000D-01	6	0.000000000D-01
7	0.000000000D-01	8	0.000000000D-01	9	0.000000000D-01	10	0.000000000D-01	11	0.000000000D-01	12	0.000000000D-01
13	0.000000000D-01	14	0.000000000D-01	15	0.000000000D-01	16	0.000000000D-01	17	0.000000000D-01	18	0.000000000D-01
19	0.000000000D-01	20	0.000000000D-01	21	0.000000000D-01	22	0.000000000D-01	23	0.000000000D-01	24	0.000000000D-01
25	0.000000000D-01	26	0.000000000D-01	27	0.000000000D-01	28	0.000000000D-01	29	0.000000000D-01	30	0.000000000D-01
31	0.000000000D-01	32	0.000000000D-01	33	0.000000000D-01	34	0.000000000D-01	35	0.000000000D-01	36	0.000000000D-01
37	0.000000000D-01	38	0.000000000D-01	39	0.000000000D-01	40	0.000000000D-01	41	0.000000000D-01	42	0.000000000D-01
43	0.000000000D-01	44	0.000000000D-01	45	0.000000000D-01	46	0.000000000D-01	47	0.000000000D-01	48	0.000000000D-01
49	0.000000000D-01	50	0.000000000D-01	51	0.000000000D-01	52	0.000000000D-01	53	0.000000000D-01	54	0.000000000D-01
55	0.000000000D-01	56	0.000000000D-01	57	0.000000000D-01	58	0.000000000D-01	59	0.000000000D-01	60	0.000000000D-01
61	0.000000000D-01	62	0.000000000D-01	63	0.000000000D-01	64	0.000000000D-01	65	0.000000000D-01		

E N E R G Y B U D G E T AFTER TIME STEP 1, IN (ENERGY/SECOND)

7.0335721D+05	NET RATE OF INCREASE(+)/DECREASE(-) OF ENERGY IN FLUID
1.4975353D+06	NET RATE OF INCREASE(+)/DECREASE(-) OF ENERGY IN SOLID GRAINS
0.00000000-01	NET ZERO-ORDER PRODUCTION(+)/LOSS(-) OF ENERGY IN FLUID
0.00000000-01	NET ZERO-ORDER PRODUCTION(+)/LOSS(-) OF ENERGY IN SOLID GRAINS
1.3068750+06	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH FLUID SOURCES AND SINKS
-4.6531506-112	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED PRESSURE
0.00000000-01	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH ENERGY SOURCES AND SINKS

ENERGY SOURCES OR SINKS AT FLUID SOURCES AND SINKS

NODE	SOURCE(+)/SINK(-) (ENERGY/SECOND)
1	6.5343750D+05
2	6.5343750D+05

ENERGY SOURCES OR SINKS DUE TO FLUID INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED PRESSURE

NODE	SOURCE(+)/SINK(-) (ENERGY/SECOND)
132	-2.3265753-112
131	-2.3265753-112

RESULTS FOR TIME STEP 225

TIME INCREMENT 4.02100+03 SECONDS

ELAPSED TIME : 9.04730+05 SECONDS
1.50790+04 MINUTES
2.51310+02 HOURS
1.04710+01 DAYS
1.49590+00 WEEKS
3.44030-01 MONTHS
2.86690-02 YEARS

T E M P E R A T U R E

NODE		NODE		NODE		NODE		NODE		NODE	
1	1.0000000000	2	1.0000000000	3	0.9999672600	4	0.9999672600	5	0.9999210140	6	0.9999210140
7	0.9998558360	8	0.9998558360	9	0.9997630250	10	0.9997630250	11	0.9996294310	12	0.9996294310
13	0.9994351200	14	0.9994351200	15	0.9991495570	16	0.9991495570	17	0.9987255330	18	0.9987255330
19	0.9980894810	20	0.9980894810	21	0.9971260710	22	0.9971260710	23	0.9956538950	24	0.9956538950
25	0.9933872880	26	0.9933872880	27	0.9898774960	28	0.9898774960	29	0.9844252870	30	0.9844252870
31	0.9759560060	32	0.9759560060	33	0.9628544820	34	0.9628544820	35	0.9427790310	36	0.9427790310
37	0.9125099900	38	0.9125099900	39	0.8679796760	40	0.8679796760	41	0.8047286920	42	0.8047286920
43	0.7191009770	44	0.7191009770	45	0.6103474680	46	0.6103474680	47	0.4831745300	48	0.4831745300
49	0.3491922700	50	0.3491922700	51	0.2250045400	52	0.2250045400	53	0.1261789050	54	0.1261789050
55	0.0602006590	56	0.0602006590	57	0.0239660870	58	0.0239660870	59	0.0078416620	60	0.0078416620
61	0.0020865430	62	0.0020865430	63	0.0004485440	64	0.0004485440	65	0.0000776440	66	0.0000776440
67	0.0000108140	68	0.0000108140	69	0.0000012130	70	0.0000012130	71	0.0000001100	72	0.0000001100
73	0.0000000080	74	0.0000000080	75	0.0000000000	76	0.0000000000	77	0.0000000000	78	0.0000000000
79	0.0000000000	80	0.0000000000	81	0.0000000000	82	0.0000000000	83	0.0000000000	84	0.0000000000
85	0.0000000000	86	0.0000000000	87	0.0000000000	88	0.0000000000	89	0.0000000000	90	0.0000000000
91	0.0000000000	92	0.0000000000	93	0.0000000000	94	0.0000000000	95	0.0000000000	96	0.0000000000
97	0.0000000000	98	0.0000000000	99	0.0000000000	100	0.0000000000	101	0.0000000000	102	0.0000000000
103	0.0000000000	104	0.0000000000	105	0.0000000000	106	0.0000000000	107	0.0000000000	108	0.0000000000
109	0.0000000000	110	0.0000000000	111	0.0000000000	112	0.0000000000	113	0.0000000000	114	0.0000000000
115	0.0000000000	116	0.0000000000	117	0.0000000000	118	0.0000000000	119	0.0000000000	120	0.0000000000
121	0.0000000000	122	0.0000000000	123	0.0000000000	124	0.0000000000	125	0.0000000000	126	0.0000000000
127	0.0000000000	128	0.0000000000	129	0.0000000000	130	0.0000000000	131	0.0000000000	132	0.0000000000

E N E R G Y B U D G E T AFTER TIME STEP 225, IN (ENERGY/SECOND)

4.1770564D+05	NET RATE OF INCREASE(+)/DECREASE(-) OF ENERGY IN FLUID
8.8934744D+05	NET RATE OF INCREASE(+)/DECREASE(-) OF ENERGY IN SOLID GRAINS
0.0000000D-01	NET ZERO-ORDER PRODUCTION(+)/LOSS(-) OF ENERGY IN FLUID
0.0000000D-01	NET ZERO-ORDER PRODUCTION(+)/LOSS(-) OF ENERGY IN SOLID GRAINS
1.3068750D+06	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH FLUID SOURCES AND SINKS
-6.1798542D-55	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED PRESSURE
0.0000000D-01	NET GAIN(+)/LOSS(-) OF ENERGY THROUGH ENERGY SOURCES AND SINKS

ENERGY SOURCES OR SINKS AT FLUID SOURCES AND SINKS

NODE	SOURCE(+)/SINK(-) (ENERGY/SECOND)
1	6.5343750D+05
2	6.5343750D+05

ENERGY SOURCES OR SINKS DUE TO FLUID INFLOWS OR OUTFLOWS AT POINTS OF SPECIFIED PRESSURE

NODE	SOURCE(+)/SINK(-) (ENERGY/SECOND)
132	-3.0899271D-55
131	-3.0899271D-55

*** LAST SOLUTION HAS BEEN STORED ON UNIT 66 ***

O B S E R V A T I O N N O D E D A T A

		NODE 34		NODE 52		NODE 64		NODE 72	
TIME STEP	TIME(SEC)	PRESSURE	TEMPERATURE	PRESSURE	TEMPERATURE	PRESSURE	TEMPERATURE	PRESSURE	TEMPERATURE
1	4.02100D+03	2.34430D+06	1.42872D-07	1.69042D+06	1.43923D-18	1.31481D+06	3.80382D-29	1.07900D+06	1.35142D-37
45	1.80945D+05	2.34430D+06	2.53780D-01	1.69042D+06	6.15373D-06	1.31481D+06	5.06001D-13	1.07900D+06	1.74037D-19
90	3.61890D+05	2.34430D+06	6.59811D-01	1.69042D+06	2.08077D-03	1.31481D+06	1.00897D-08	1.07900D+06	4.20084D-14
135	5.42835D+05	2.34430D+06	8.46375D-01	1.69042D+06	2.63662D-02	1.31481D+06	1.81630D-06	1.07900D+06	4.18606D-11
180	7.23780D+05	2.34430D+06	9.26530D-01	1.69042D+06	1.01824D-01	1.31481D+06	4.75215D-05	1.07900D+06	4.04398D-09
225	9.04725D+05	2.34430D+06	9.62854D-01	1.69042D+06	2.25005D-01	1.31481D+06	4.48544D-04	1.07900D+06	1.09754D-07

SUTRA SIMULATION TERMINATED AT COMPLETION OF TIME STEPS
***** ********** ********** ** ********** ** **** *****