PNNL-11218
UC-2010

STOMP
Subsurface Transport Over Multiple Phases

User's Guide

M. D. White
M. Oostrom

October 1997

Prepared for the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

This work is funded by the Office of Science and Technology, within the U.S. Department of Energy's Office of Environmental Management, under the Plumes Focus Area.

Pacific Northwest National Laboratory
Richland, Washington 99352

DISCLAIMER

Preface

This guide describes the general use, input file formatting, compilation and execution of the STOMP (Subsurface Transport Over Multiple Phases) simulator, a scientific tool for analyzing multiple phase subsurface flow and transport. Examples of the STOMP simulator applied to classical groundwater problems are provided in a companion application guide. A description of the simulator's governing equations, constitutive functions and numerical solution algorithms are provided in a companion theory guide. In writing these guides for the STOMP simulator, the authors have assumed that the reader comprehends concepts and theories associated with multiple-phase hydrology, heat transfer, thermodynamics, radioactive chain decay, and nonhysteretic relative permeability-saturation-capillary pressure constitutive functions. The authors further assume that the reader is familiar with the computing environment on which they plan to compile and execute the STOMP simulator.

The STOMP simulator requires an ANSI FORTRAN 77 compiler to generate an executable code. The memory requirements for executing the simulator are dependent on the complexity of physical system to be modeled and the size and dimensionality of the computational domain. Likewise execution speed depends on the problem complexity, size and dimensionality of the computational domain, and computer performance. One-dimensional problems of moderate complexity can be solved on conventional desktop computers, but multidimensional problems involving complex flow and transport phenomena typically require the power and memory capabilities of workstation or mainframe type computer systems.

Summary

The U. S. Department of Energy, through the Office of Technology Development, has requested the demonstration of remediation technologies for the cleanup of volatile organic compounds and associated radionuclides within the soil and groundwater at arid sites. This demonstration program, called the VOC-Arid Soils Integrated Demonstration Program (Arid-ID), has been initially directed at a volume of unsaturated and saturated soil contaminated with carbon tetrachloride, on the Hanford Site near Richland, Washington. A principal subtask of the Arid-ID program involves the development of an integrated engineering simulator for evaluating the effectiveness and efficiency of various remediation technologies. The engineering simulator's intended users include scientists and engineers who are investigating soil physics phenomena associated with remediation technologies. Principal design goals for the engineer simulator include broad applicability, verified algorithms, quality assurance controls, and validated simulations against laboratory and field-scale experiments. An important goal for the simulator development subtask involves the ability to scale laboratory and field-scale experiments to full-scale remediation technologies, and to transfer acquired technology to other arid sites. The STOMP (Subsurface Transport Over Multiple Phases) simulator has been developed by the Pacific Northwest National Laboratory[a] for modeling remediation technologies. Information on the use, application, and theoretical basis of the STOMP simulator are documented in three companion guide manuals. This manual, the Theory Guide, provides theory and discussions on the governing equations, constitutive relations, and numerical solution algorithms for the STOMP simulator.

The STOMP simulator's fundamental purpose is to produce numerical predictions of thermal and hydrogeologic flow and transport phenomena in variably saturated subsurface environments, which are contaminated with volatile or nonvolatile organic compounds. Auxiliary applications include numerical predictions of solute transport processes including radioactive chain decay processes. Quantitative predictions from the STOMP simulator are generated from the numerical solution of partial differential equations that describe subsurface environment transport phenomena. Description of the contaminated subsurface environment is founded on governing conservation equations and constitutive functions. Governing coupled flow equations are partial differential equations for the conservation of water mass, air mass, volatile organic compound mass, thermal energy. Constitutive functions relate primary variables to secondary variables.

---

[a] Pacific Northwest National Laboratory is operated by Battelle Memorial Institute for the U.S. Department of Energy under Contract DE-AC06-76RLO 1830.

Solution of the governing partial differential equations occurs by the integral volume finite difference method. The governing equations that describe thermal and hydrogeological flow processes are solved simultaneously using Newton-Raphson iteration to resolve the nonlinearities in the governing equations. Governing transport equations are partial differential equations for the conservation of solute mass. Solute mass conservation governing equations are solved sequentially, following the solution of the coupled flow equations, by a direct application of the integral volume finite difference method. The STOMP simulator is written in the FORTRAN 77 language, following American National Standards Institute (ANSI) standards. The simulator utilizes a variable source code configuration, which allows the execution memory and speed be tailored to the problem specifics, and essentially requires that the source code be assembled and compiled through a software maintenance utility.

KEYWORDS: subsurface, porous media, multiple phase, groundwater, nonaqueous phase liquid (NAPL), volatile organic compound (VOC), variably saturated, frozen-soil conditions, brines, nonequilibrium kinetics, solute transport, radioactive chain decay, hysteretic, fluid entrapment, finite-difference, Newton-Raphson, nonlinear, modeling, TVD transport, banded linear system solver, conjugate gradient solver, unsymmetric multifrontal package.

Acknowledgements

This work was funded by the Office of Science and Technology, within the Department of Energy's Office of Environmental Management, under the Plumes Focus Area, with the support of the following contributors:

Headquarters:	Office of Science and Technology David Biancosino
Focus Area/Program:	Plumes Focus Area James A. Wright
Operations Office:	Richland Operations Office Science and Technology Programs Division Jeffrey A. Frey, Technical Program Officer
Contractor:	Pacific Northwest National Laboratory Environmental Management Technology Development Program Steven C. Slate, Manager

Financial assistance for the technical review of this document was provided by the Office of Nuclear Regulatory Research of the U.S. Nuclear Regulatory Commission (NRC) under project JCN W6503. STOMP has been used by PNNL staff to support NRC projects JCN W6503 and JCN L2466.

Contents

Figures

Tables

1.0 Introduction

The VOC-Arid Soils Integration Demonstration Program, abbreviated as Arid-ID and funded by the U.S. Department of Energy (DOE), Office of Technology Development (OTD), is directed at the cleanup of volatile organic compounds and associated radionuclides and heavy metals in soils and groundwater at arid sites. The initial demonstration site is located within the 200 West Area on the Hanford Site near Richland, Washington. The site contains a volume of soil contaminated with carbon tetrachloride, which includes approximately 200 vertical feet of contaminated unsaturated sediments underlying inactive disposal sites and overlying a 7-square-mile plume of contaminated groundwater. A critical component of the Arid-ID program involves assessing the impact of spatial heterogeneity of subsurface materials on remediation processes and evaluating the effectiveness and efficiency of demonstrated remedial technologies. Because of the complexity of subsurface flow and heat transport phenomena, these assessments and evaluations will require complex numerical tools for their completion. Numerical tools allow scientists and engineers to integrate the current knowledge of contaminant behavior in the subsurface environment to predict and evaluate the performance of proposed remediation methods against established technologies.

A principal subtask of the Arid-ID program involves the development of an engineering simulator (numerical tool), which is capable of numerically simulating proposed remediation processes. The design goals are that the engineering simulator: 1) be accessible and exploitable to scientists and engineers familiar with subsurface environment phenomena, but not necessarily numerical modeling technicalities, 2) have enough general applicability to recruit a user group that is capable of supporting training, maintenance, and enhancement activities, 3) be verified by comparisons to analytical solutions and benchmarked against existing simulators, 4) be validated against germane laboratory and field experiments, and 5) have controlled configuration and documentation under an appropriate quality assurance program. An engineering simulator named STOMP, an acronym for Subsurface Transport Over Multiple Phases, has been developed by the Pacific Northwest National Laboratory[a] which achieves the five design goals described above. This document, one of three companion documents, has been written to provide users of the STOMP simulator with necessary information for selecting an appropriate operational mode,

---

(a) Pacific Northwest National Laboratory is operated by Battelle Memorial Institute for the U.S. Department of Energy under Contract DE-AC06-76RLO 1830.

understanding the code flow path and design, creating input files, dimensioning the executable, compiling and executing, and interpreting simulation outputs. The second companion document, the STOMP Applications Guide (Nichols et al. 1997), provides users of the STOMP simulator with applications of the simulator to classical groundwater problems. The third companion document, the STOMP Theory Guide (White and Oostrom 1996), provides users of the STOMP simulator with information about the solved governing and constitutive equations, numerical algorithms, and solution techniques.

This guide has been organized in a manner which duplicates the normal approach to solving a subsurface flow and transport problem with the STOMP simulator. The STOMP simulator has been designed with a variable source code, where source code configurations are referred to as operational modes. Operational modes are classified according to the solved governing flow and transport equations and constitutive relation extensions. Therefore, prior to creating an input file or assembling the source code the user must choose the appropriate operational mode for the particular subsurface system of interest. The selection of an operational mode requires that the user conceptualize the physical system as a computational system, which will always require making simplifying assumptions about the physical system. The complexity and execution speed of an operational mode is generally inversely related to the number of simplifying assumptions the user can justify about the physical system. Regardless of the operational mode complexity, the simulation of physical subsurface systems with the STOMP simulator always involves assumptions inherent to the founding governing equations and constitutive relations.

Having chosen an operational mode, the user should then set about creating an input file. Input files, simply stated, are translations of the physical system description into a computational system. The user communicates information about the physical system to the STOMP simulator through the input file. The input file, operational mode, and linear system solver determine the source code configuration and required dimension of the executable. Executable dimensions are controlled through parameters, which are communicated to the STOMP simulator through *the "parameters"* file. It is the user's responsibility to ensure that the parameters listed in the *"parameters"* file are sufficiently dimensioned for the input file, operational mode, and selected linear system solver. Having created "input" and "parameters" files that describe the physical system to be modeled, the next step in the simulation process involves assembling the source code, creating an executable, and executing the code.

As previously mentioned, the STOMP simulator has a variable source code configuration, which is dependent on the selected operational mode. The source coding for each operational mode comprises mode specific coding and global coding from a library of routines. Assembly and compilation of the source coding for a particular operational mode has been facilitated through a UNIX utility called "make" (Talbott 1988). Instructions for using these utilities are described in this document. The STOMP simulator offers the user considerable flexibility in executing and controlling the execution of simulations. The user can create numerous execution periods within a single simulation, which differ with respect to time step advancement and convergence control. Controls of this sort are beneficial for simulations involving complex time-varying boundary conditions or sources. Moreover, the user can produce intermediate *"restart"* files which are essentially "snapshots" of the primary variables. Simulations can be restarted from intermediate *"restart"* files using the altered or unaltered descriptions of the computational system. Various techniques and approaches for executing the simulator are also described in this document. Other examples of simulator applications and application techniques can be found in the STOMP Application Guide (Nichols et al. 1997).

The STOMP simulator generates results in a number of different formats, which can be controlled by the user. This guide describes these formats and the output capabilities of the simulator. Output formats include text files of simulation results and printing to the standard input/output device, which is typically a terminal or workstation screen. All input and output files generated by the simulator including *"restart"* files are ASCII text files, which can be read or modified by conventional text editors. Output data has been grouped in two basic formats, historic and snapshot. Historic data are records of selected variables over the simulation period. These data records are essential for creating graphs or plots showing the progression of variable values over simulation time. Snapshot data are records of selected variables at a particular moment in time over the computation domain. These data records are essential for generating images of variable values over the computational space. Snapshot data records can also be linked in a time sequence to show the evolution of variable values in both computational time and space.

Beyond providing operational type instructions, this document also contains useful information for executing the simulator with a debug utility or making modifications to the source code. The user may often be able to isolate input errors or convergence problems by tracing the execution of the simulator with a debug utility. To assist the user in performing debug executions or in making modifications to the code, critical information about the operational flow path, subroutine functions, and variable descriptions have been provided in this guide. The source

coding for the STOMP simulator is suitably documented with comment statements, however none of the variable or parameter names are defined nor is a general flow path for simulator outlined. Therefore, this guide provides essential information about subroutines and variables. Indexing of primary and secondary field variables and flux variables in the STOMP simulator is complex and depends on the operational mode. A description of the indexing patterns developed for the simulator are also provided in this guide.

2.0 Fundamentals

2.1 Introduction

The STOMP simulator has been designed to solve a wide variety of nonlinear, multiple-phase, flow and transport problems for variably saturated geologic media. Partial differential conservation equations for component mass, energy, and solute mass comprise the fundamental equations for the simulator. Coefficients within the fundamental equations are related to the primary variables through a set of constitutive relations. The conservation equations for component mass and energy are solved simultaneously, whereas the solute transport equations are solved sequentially after the coupled flow solution. The variable source code configuration allows the user to select the combination of solved fundamental equations. The current version of the STOMP simulator recognizes ten coupled flow equation combinations. Each coupled flow equation combination is referred to as an operational mode and may additionally include the solution of a number of transported solutes. The associated constitutive relations for each recognized operational mode are automatically incorporated into the source code as required.

In reading this document, it is important to distinguish between phases and components. Phases are composed of components. The terms aqueous, gas, nonaqueous phase liquid (NAPL), ice, and solid will be used exclusively in referring to phases. The terms water, air, oil, salt, and surfactant refer to components. The aqueous phase is primarily composed of liquid water with lesser amounts of dissolved oil, air, salt, and surfactant. The gas-phase composition can be highly variable, but can contain air, water vapor and oil vapor. The NAPL phase is assumed to be composed only of liquid oil with no dissolved air or water. The ice phase is composed of pure frozen water. The current version of the STOMP simulator permits only a single oil component within the NAPL phase (i.e., a noncompositional model). Dissolved oil and transported solutes can sorb to the solid phase (i.e., the solid rock/soil matrix).

The STOMP simulations are limited in application scope according to the solved fundamental equations, the associated constitutive theory, inherent assumptions, computer execution speed and memory, and the user's creativity. A critical component to correct application of the STOMP simulator and comprehension of output results requires an understanding by the user of the assumptions taken to develop the various flow and transport algorithms. The simulator

is capable of predicting flow and transport behavior for a variety of subsurface systems, however, application of the simulator to problems which violate an inherent assumption in the simulator's design or fundamental equations could yield incorrect results. STOMP is principally limited to flow through variably saturated porous media, which can be characterized with an extended form of Darcy's law. Additionally, the simulator is limited to a maximum of three immiscible phases referred to as the aqueous, nonaqueous liquid, and gas phases. Low solubilities are assumed for the liquid phases and interphase mass transfer assumes equilibrium conditions. Specific to the energy equation, the principal assumptions are that heat transport by gas-phase conduction and the kinetic nature of thermodynamic processes are neglected. Solute transport solutions are computed sequentially to the coupled flow equations. This approach requires the assumption that solutes are passive scalars with respect to the flow equations, which is equivalent to assuming solute concentrations are dilute.

The STOMP simulator solves transient flow and transport problems in the subsurface environment in one, two, or three dimensions. Coordinate systems must be orthogonal and currently are limited to Cartesian, tilted Cartesian, and cylindrical, where the vertical coordinate of the cylindrical system must be aligned with the gravitational vector. The STOMP simulator solves steady-state problems either directly or through false-transients starting from a user specified initial state. Direct solutions to steady-state problems are possible for initial conditions sufficiently close to the solution, therefore, transient solutions to steady-state conditions are the recommended approach. All boundary conditions, sources, and sinks are time variant and allow the user considerable control over transient simulations. Hydrogeologic properties can be spatially varied throughout the computational domain, within the resolution of a node volume. Hydraulic and thermal transport properties for the porous medium can be anisotropic, where the diagonal tensor elements are assumed to be aligned with the principal axes of the coordinate system. Coupled flow solutions can be obtained for selected one-, two-, or three-phase systems under isothermal or nonisothermal conditions. Transport of radioactive solutes with chain-decay tracking can be coupled to all transport solutions with the assumption of dilute solute concentrations. A variety of boundary conditions are available for each operational mode, which may be applied selectively over the boundary surfaces. Unspecified boundary surfaces are always assumed to be zero flux surfaces. Inactive nodes can be specified by the user and boundary conditions can be applied to surfaces separating active (computational) and inactive (noncomputational) nodes. Selected operational modes allow the user to invoke a dynamic domain option which temporarily sets quiescent nodes to an inactive set, thus removing them from the computational domain.

The fundamental coupled flow equations are solved following a integral volume finite-difference approach with the nonlinearities in the discretized equations resolved through Newton-Raphson iteration. Linear systems which result from the Newton-Raphson linearization or the solute transport solution can be solved with a direct banded matrix solver, an unsymmetric-pattern multifrontal package or an indirect conjugate gradient based solver. All these linear system solvers are commercially available software products that have been adapted for the particulars of the STOMP simulator. The STOMP simulator allows considerable control over simulation parameters related to convergence, time stepping, solution techniques, and execution limits. A single simulation can be divided into multiple execution periods, each with a different set of solution control parameters. Restart capabilities have also been included in the simulator, which can resume a simulation from user define points with or without alterations to input parameters. Output from the simulator can be totally controlled by the user and is written both to files and to the standard input/output device (e.g., screen). Output forms included time histories of selected variables, time "snap shots" of selected variables across the computational domain, and variable integrals for sources and fluxes across boundary and internode surfaces.

The dominant nonlinear functions within the STOMP simulator are the relative permeability-saturation-capillary pressure (k–S–P) relations. The STOMP simulator allows the user to specify these relations through a large variety of popular and classic functions. Two-phase (water-air) k–S–P relations can be specified with hysteretic or nonhysteretic functions or nonhysteretic tabular data. Entrapment of air with imbibing water conditions can be modeled with the hysteretic two-phase k–*S*–*P* functions. Two-phase *k*–*S*–*P* relations span both saturated and unsaturated conditions. The aqueous phase is assumed to never completely disappear through extensions to the *S*–*P* function below the residual saturation and a vapor pressure lowering scheme. Three-phase (aqueous-NAPL-gas) *k*–*S*–*P* relations can be specified with hysteretic or nonhysteretic functions. Tabular three-phase *k*–*S*–*P* relations are not currently available. Entrapment of nonwetting fluids (i.e., gas in aqueous, gas in NAPL, and NAPL in aqueous) with imbibing wetting fluid conditions can be modeled with the hysteretic three-phase *k*–*S*–*P* functions. The wettability order for fluids in the STOMP simulator is assumed to follow the descending order aqueous to NAPL to gas. Three-phase *k*–*S*–*P* relations span both total-liquid saturated and unsaturated conditions and aqueous saturated and unsaturated conditions. Phase appearances and disappearances are possible for the gas and NAPL phases, however, the aqueous phase is assumed to never completely disappear through extensions to the *S*–*P* function below the residual saturation and a vapor pressure lowering scheme.

2.2 Operational Modes

The STOMP simulator operational modes do not comprise all of the possible combinations of coupled governing equations. The combination sets of coupled governing equations selected for inclusion in the list of operational modes represent those with the greatest utility for physical systems. For example, a two-phase nonvolatile nonisothermal operational mode could be envisioned, which solved the water mass and energy conservation equations. This operational mode would invoke the assumption of a nonparticipating gas phase. Two-phase flow and transport through porous media under thermal gradients, however, strongly depends on the diffusion, dispersion, and advection transport through gas phase, even for low thermal gradients. Therefore, this operational mode while capable of functioning and producing converged solutions would have limited utility, because its associate premise that gas phase transport could be neglected. Other combinations of governing equation sets with associated assumptions have considerable utility for specific systems or problems and have been coded, but have not been selected for inclusion in the STOMP guides. For example, an operational mode has been created that solves the water mass, air mass, oil mass, and/or energy equation with the assumption that oil concentrations remain below the aqueous solubility limit. This operational mode differs from one which models the oil as a dilute solute, because the oil mass equation is solved simultaneously with the other coupled flow equations and fluid properties have a dependence on the oil concentration. This operational mode has utility in the investigation of the remediation of oil contaminants in dissolved plumes.

Operational modes are indentified according to the coupled conservation equations that are solved. All operational modes support solute transport calculations. The solved conservation equations and primary assumptions are summarized in Table 2-1 for each operational mode. Beyond the set of solved coupled equations, some of the operational modes have unique features developed for specific applications, which should be noted. The Water operational mode contains an optional dynamic domain feature. This feature converts relatively invariant nodes into noncomputational or inactive nodes, therefore improving execution performance. With this feature all active nodes are included in the computational set during the initial iteration. If the residual for the water mass conservation equation for a particular node is less than a specified limit then that node becomes an inactive node for the remainder of iterations within the time step. The Water operational mode additionally contains two algorithms for computing the partitioning of transported solutes between the solid and aqueous phases. The first approach assumes that the solid surface is continuously wet independent of the aqueous saturation; whereas, the second approach assumes

that the wetted surface is proportional to the aqueous saturation. This operational mode also includes a feature for solute dependent retardation. With this feature solid-aqueous partition coefficient for one solute is dependent on the concentration of another solute. The feature allows the modeling of surfactant type reactions between solvents. Other less used features of this operational mode are the screened outlet well The Water-Air operational mode contains a unique hysteretic air entrapment constitutive theory that allows dissolution and expansion of entrapped air. This feature was developed specifically to examine mechanisms for entrapped gas release from a declining water table. The Water-Air-Energy operational mode contains models for computing ice saturations under soil freezing conditions. This feature does not include soil stress processes necessary to compute mounding and heaving. This operational mode addtionally contains a feature to correct water vapor pressures for vapor pressure lowering through capillary forces and aqueous saturations less than the residual saturation can occur through application of an extension to the capillary pressure-saturation constitutive model. Enhanced vapor diffusion from thermal gradients can be considered as a function of the thermal gradient, moisture content, soil-moisture retention function, and soil clay fraction.

Table 2.1 Operational Mode Summary

Operational Mode w/ Options	Solved Coupled Equations	Primary Assumptions
Water	water mass	isothermal conditions passive gas phase no NAPL phase no dissolved oil no brine local thermodynamic equilibrium
Water-Air	water mass air mass	isothermal conditions no NAPL phase no dissolved oil no brine local thermodynamic equilibrium
Water-Air-Energy w/ Ice	water mass air mass thermal energy	no NAPL phase no dissolved oil no brine local thermodynamic equilibrium
Water-Oil	water mass oil mass	isothermal conditions single component NAPL phase passive gas phase no brine local thermodynamic equilibrium
Water-Oil-Air	water mass oil mass air mass	isothermal conditions single component NAPL phase no brine local thermodynamic equilibrium
Water-Oil-Air-Energy	water mass oil mass air mass thermal energy	single component NAPL phase no brine local thermodynamic equilibrium
Water-Oil-Dissolved Oil	water mass oil mass dissolved oil mass	isothermal conditions single component NAPL phase passive gas phase no brine kinetic oil dissolution local thermodynamic equilibrium

Table 2.1 (Contd.)

Operational Mode w/ Options	Solved Coupled Equations	Primary Assumptions
Water-Oil-Dissolved Oil-Surfactant	water mass oil mass dissolved oil mass surfactant mass	isothermal conditions single component NAPL phase passive gas phase no brine kinetic oil dissolution local thermodynamic equilibrium surfactant enhanced dissolution and mobilization
Water-Salt	water mass salt mass	isothermal conditions passive gas phase no NAPL phase no dissolved oil local thermodynamic equilibrium
Water-Air-Salt	water mass air mass salt mass	isothermal conditions no NAPL phase no dissolved oil local thermodynamic equilibrium
Water-Air-Energy-Salt w/ Ice	water mass air mass salt mass thermal energy	no NAPL phase no dissolved oil local thermodynamic equilibrium

3.0 Code Design

3.1 Introduction

The primary design guides for the STOMP simulator have been modularity, computational efficiency, and readability. A modular code architecture is beneficial because of the ease of reading, maintaining, and modifying the algorithms and is essential to the variable configuration source code. Computational efficiency refers to both memory requirements and execution speed. The STOMP simulator has been designed with a variable configuration source code which allows the memory requirements and code algorithms to be partially customized to the computational problem. This approach offers considerable advantages with respect to achieving a computationally efficient code design. Within this source code framework, however, many design choices have been made that affect computational efficiency. Algorithm design often offers options between memory and speed. For example, to lessen memory requirements a code designer may opt to repeatedly compute commonly used variables. Conversely, execution speed may be increased at the cost of increased memory requirements, by storing commonly used variables after their initial computation. Generally, the approach in the STOMP simulator has been to favor increased memory requirements to gain computational speed. This design approach has been chosen because of current state of computer architecture and capabilities. Because the STOMP simulator has been created as a scientific tool, algorithm readability has been an primary design guide. As a scientific tool, the simulator was never expected to remain unmodified, but rather a constantly changing package of software tools which could be applied to new or more complex problems. This design goal makes readability an essential feature of the code. Code readability has been achieved through an extensive use of comments, a modular design, a large group of common blocks, and minimal subroutine and function arguments.

3.2 Flow Path

The general flow path for all operational modes of the STOMP simulator comprises three components, initialization, iteration, and closure. A flow chart for the initialization, iteration, and closure components of the main program is shown in Figure 3.1, where the enclosing boxes indicate either a single or group of routines. The initialization component of the program is

executed once during a simulation. The routines in the initialization component are executed in sequence shown in Figure 3.1, from the program start to the start of the first time step. The iteration component of the program contains a pair of nested loops, an outer loop for time stepping and an inner loop for Newton-Raphson linearization. Termination of the Newton-Raphson loop occurs with a successful convergence or after an iteration limit violation. Termination of the time-stepping loop after a simulation limit or a time-step reduction limit violation has occurred. Regardless of the cause for termination during the time-stepping loop the closure routines are executed at the simulation completion. The transport solution is shown as a single routine on the STOMP flow diagram. It comprises, however, several transport routines within a solute loop. The flow diagram for the solute transport portion of the iteration component is shown in Figure 3.2.

The initialization component (Initialize Variables) of the program flow path begins by initializing all common variables. All common variables, those in the common blocks of the *"commons"* file, are initialized at this point either to zero or with default values. Unless specifically defaulted, integers are initialized to 0, real variables are initialized to 0.D+0, and character strings are initialized to blank. During the variable initializations the *"input"* and *"output"* files are opened. The next routine (Print File Banners) prints the welcome statement, disclaimer, and banner to the standard output device (screen) and the *"output"* file. This is followed with the procedures (Read Input File) for reading the *"input"* file. The *"input"* file is read using a predefined card order. After each card has been read, the *"input"* file is rewound and searched from the beginning for the next card to be read. This approach allows the user to sequence input cards randomly within the *"input"* file. The card read order defined within the code software is critical and should not be altered. When appropriate the *"restart"* file is read for input data information during these procedures. Input data are then checked for saturation or thermodynamic state consistency (Check Physical States). If an error is found in these routines the simulation will be terminated with an associated error message. If no errors in state conditions are noted then the program continues the initializing process with the Jacobian matrix pointers (Set Matrix Pointers).

The Jacobian matrix structure varies with operational mode and grid geometry, refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of the numerical methods and linear system solvers. Jacobian matrix pointers are integer arrays that relate primary variables for a particular node to a location in the Jacobian matrix. If the simulation does not involve dynamic domains then these pointers are constants during the simulation and need to be computed only once. The next three initialization routines (Saturation Properties, Physical Properties and Solute Concentrations) compute initial values for the secondary variables from the initial conditions

specified through the *"input"* or *"restart"* files. These routines have been divided into three components those for computing phase saturation related variables, thermodynamic properties, and solute concentrations.

The iteration component of the program flow path contains a pair of nested loops. The outer loop increments time and represents a single time step and the inner loop increments iterations of the Newton-Raphson linearization technique. During a single time increment loop (time step) both the flow and transport governing equation sets are solved. The Newton-Raphson linearization loop is applicable only to the solution of the coupled mass and heat flow governing equations. The solute transport governing equations are solved directly (without iteration) and sequentially to the iterative flow solution (Start Time Step). Each time step loop starts with a computation of the new time step and increments to the time and time step counter. Time steps are computed with an algorithm based on the most previous complete time step, the time step acceleration factor, and the time to a transition point. Transition points occur with changes in execution period times, boundary condition times, source times, and output times. Time steps always conclude on transition points, which commonly requires the time step to be temporarily reduced. The time step following a temporary reduction to meet a transition point time will resume the prior time stepping levels, except in the case of execution period transitions where a new initial or maximum time step has been declared. The next procedure (Load Old Time Arrays) involves loading the previous time step arrays for field variables. Field variables from the array location for the current field variable value are loaded into the array location for the previous time step field variable value. The next three routines (Reference Node Output, Write Plot File and Write Restart File) prior to starting a Newton-Raphson iteration loop involves writing previous time step results to output. Simulation results are written to *"output"* file the standard input/output device (screen), a *"plot.n"* file, and a *"restart.n"* file, depending on the directives made by the user on the "Output Control" Card.

The Newton-Raphson iteration loop solves the coupled governing flow equations for component mass and energy. The first procedure (Boundary Properties) within this loop involves the calculation of boundary surface properties. Each declared boundary surface, has associated field variables which are computed with the same algorithms as the node field variables, but only when the boundary condition is active. This approach eliminates the necessity for computing field variable values for disabled boundary surfaces. The next group of routines (Interior Fluxes) computes fluxes across interior surfaces (those surfaces between active nodes). Fluxes which are computed within these routines are dependent on the operational mode and include Darcy phase velocities, component diffusion-dispersion fluxes, thermal conductive flux, thermal advective flux,

and thermal diffusion-dispersion fluxes. The same group of flux variables is computed in the next procedure (Boundary Fluxes) for each active boundary surface. As with the procedure for computing field variables on boundary surfaces the approach of computing flux variables only for active boundary surfaces eliminates computing unused boundary flux variables. The next procedure (Source Contributions) computes source and/or sink contributions to each of the governing flow equations from the user specified inputs on the "Source" Card. Source contributions are stored in arrays and subsequently used in computing the Jacobian matrix coefficients. At this point in the Newton-Raphson iteration loop, all field and surface flux variables have been computed, along with the source contributions. These variables compose the primary components of the governing flow equations.

The Jacobian matrix coefficients and solution vector are computed in a multiple stage sequence. The first stage involves setting all of the previous coefficient arrays to zero. This stage is necessary because the nonzero elements of the Jacobian matrix will change with time step as various boundary condition transition between active and inactive states. The second stage (Load Jacobian Matrix) involves computing the Jacobian matrix and solution vector with the assumption of zero flux boundary conditions for all boundary surfaces, including surfaces between active and inactive nodes. The Jacobian matrix loading procedure depends on the operational mode, but is sequenced according to governing partial differential equations. Coefficients for the water mass conservation equation are loaded first, followed by the air mass, oil mass, and energy conservation equations. The resulting system of equations represents the discretized and linearized system of governing flow equations with zero flux boundary conditions imposed, where the source contributions have been incorporated. The final stage (Boundary Matrix Modify) modifies this linear system according to the active user imposed boundary conditions. Boundary conditions will alter both the coefficient matrix and solution vector. With the Jacobian matrix and solution vector elements computed, the next procedure (Solve Linear System) involves solving the linear system of equations. The linear system is solved either with a direct banded matrix solver or a iterative conjugate gradient solver. Both routines return corrections to the primary variables in the solution vector array.

Corrections to the primary variables, computed from the linear system solvers, are used to update the primary variables and determine convergence. The Newton-Raphson procedure computes corrections to the primary variable set with each iteration. The starting values for primary variables for each new time step are the previous time step values of the primary variables, as these values represent reasonable estimates of the future values. For a convergent iteration scheme, each successive iteration yields diminishing corrections to the primary variables. Phase

transitions and primary variable switching schemes, however, can yield temporary increases in the correction to a particular primary variable. The two procedures (Update Primary Variables and Compute Convergence) that immediately follow the linear system solver procedures update the primary variables and determine convergence. Convergence occurs if the normalized values of the primary variable corrections for all unknowns falls below a user-defined value (typically 10^{-6}). The next three (Primary Increments, Saturation Properties and Physical Properties) are executed independent of the convergence result. If convergence occurs (Convergence Switch) then these computed values represent the current values of secondary variables at the conclusion of the time step; otherwise they represent the current iterate values of the secondary variables. If convergence occurs then the solution procedure continues with solute transport procedures. At the conclusion of nonconvergent iterations two additional checks are made. If the iteration count does not exceed the user specified limit then program proceeds with a new Newton-Raphson iteration loop. If the iteration count exceeds the limit (Iteration Limit?), then a check is made on the count of successive time step reductions. If convergence has failed and resulted in a time step reduction, four times in succession (Time Reduction Limit?) then the simulation aborts and program execution is transferred to the closure routines. Otherwise, the time step is reduced (Reduce Time Step), the program execution is transferred to the beginning of the time increment loop, and another attempt is made reach a converged solution for the time step.

The transport solution procedure follows the iterative solution of the mass and heat flow equations and, although a direct solution scheme, involves looping over the number of solutes. A procedure flow diagram for the transport solution routines is shown in Figure 3.2. Prior to entering the solute loop the interior-surface and boundary-surface flux procedures are called to obtain values of all flux variables at the conclusion of a time step. Flux values at the conclusion of a time step will typically vary from those computed during the last iteration of the current time step, because the primary and secondary variables will have been updated near the bottom of the last Newton-Raphson iteration loop. The surface flux calculations within the Newton-Raphson loop differ from these surface calculations in that only the current unincremented value of the surface flux is computed. Refer to Section 3.4 for a description of surface flux variable arrays.

The transport solution procedure loops over the number of solutes in the reverse order that they are defined on the "Solute/Fluid Interactions" Card. A reverse looping order is used to compute progeny solutes prior to computing parent solutes. This approach allows sequential coupling between solutes that decay radioactively or chemically with first order reaction rates yielding solute products. The first step (Solute Partitioning) of the transport solution loop involves computing the equilibrium distribution of solute between the fluid and solid phases. The second

step (Zero Jacobian Matrix) is to initialize the coefficient matrix and solution vector elements to zero. As with the flow solution scheme the possibility for boundary conditions and sources that transition makes initializing the linear system elements mandatory. The third step (Solute Sources) of the transport solution loop is to compute the solute source and/or sink contributions. Solute source contributions are incorporated directly into the coefficient matrix and solution vector elements. The fourth step (Load Jacobian Matrix) involves loading the coefficient matrix and solution vector. As with the flow solution scheme, the coefficient matrix and solution vector elements are computed assuming zero-flux conditions on all boundary surfaces, then modified for the boundary conditions (Boundary Matrix Modify). Element loading depends on the operational mode and occurs sequentially by phases, where the aqueous phase contributions are loaded first followed by the gas and NAPL phases. Solid phase contributions are loaded with the first active fluid phase. The fifth step (Solve Linear System) of the transport solution procedure is the solution of linear system of equations. Algorithms for solving the transport linear system of equations are identical to those for the flow solution, accept that the returned results for the transport solution are directly the volumetric solute concentrations. The sixth step (Update Solute Conc.) of the transport solution is to update the solute concentrations with their newly computed values. The concluding steps (Zero Solute Fluxes, Compute Solute Fluxes and Integrate Solute Sources) for the transport solution compute solute fluxes and integrate the solute sources. Once the transport solution loop has been executed for every solute the program execution returns to the flow solution procedures. Time steps for the transport solution are by default equal to those used for the flow solution. Although not currently an option with the STOMP simulator, the transport solution could be advanced in fractional values of the flow solution time steps.

The closure routines are only executed once at the successful or unsuccessful conclusion of a simulation. These routines generate final *"plot.n"* files, *"restart.n"* files, and close all opened files. Upon successful conclusion to the simulation the final *"restart.n"* file will contain a record of the primary variables at the conclusion of the final time step. Conversely, if a simulation concludes unsuccessfully, because of a convergence failure or otherwise, the *"restart.n"* file will contain a record of the primary variables at the conclusion of the previous converged time step.

3.3 Subroutines

Each operational mode of the STOMP simulator comprises global and mode dependent subroutines. Global subroutines are those subroutines which are generally included in more than one operational mode. Mode dependent subroutines, however, are associated with a single

operational mode. Subroutine names are generally descriptive abbreviations. These abbreviations frequently contain the letters G, L, or N, which respectively correspond to the gas, aqueous, or NAPL phases. Other common letters are A, O, and W, which represent air, oil and water components, respectively. The letters C and T often refer to the solute transport and energy equations. A bold letter **B** refers to a group of similar subroutines written for the bottom, south, west, east, north, and top surfaces, respectively. Each surface will have a corresponding subroutine with the subroutine name having a B, S, W, E, N, or T in place of the bold letter **B** to represent the bottom, south, west, east, north, or top surface, respectively. A short description of a subroutine's function can generally be found in the heading portion of the source coding. Mode dependent subroutine names contain a numerical suffix, which corresponds to the operational mode. Refer to Table 3.1 for operational mode index definitions. Every subroutine and function begins with a series of comment statements which includes disclaimers, a short description of the coding, a creation date stamp, and a modification date stamp. The modification date stamp is used to track the last modification date for a particular subroutine. The STOMP simulator variable structure was primarily designed around large arrays held in common blocks, which are included in nearly every subroutine. Therefore, subroutines generally have few arguments. The subroutines associated with the hysteretic saturation functions were written prior to the STOMP simulator and generally have a different programming style than other portions of the code.

3.4 Variables

The STOMP simulator has been designed with the principal variables defined in common blocks, which are included in nearly all subroutines. This approach reduces the number of arguments which are passed between routines and increases the readability of the code, because variable names remain unchanged between routines. The STOMP simulator has been coded without equivalence statements and no variables are temporarily overwritten to save memory. Variable names generally follow the intrinsic protocols for FORTRAN 77, where integer variables begin with letters in the range "I-N" and real valued variables begin with letters in the ranges "A-H" and "O-Z." Variables that begin with the letter "L" generally indicate integer parameters.

Field variables are defined at node points and are represented with two dimensional arrays. Examples of field variables include primary unknowns, saturation properties, rock/soil properties, and thermodynamic properties (e.g., temperature, pressure, phase saturation, porosity, density, and viscosity). The first index of a field variable indicates the time step or increment status. The second index of a field variable indicates the node number. The dimension of the first index

depends on the operational mode or equivalently number of coupled flow equations; where the dimension will equal the number of unknowns plus two. Therefore, field variables at each node comprise a number of elements equal to the number of unknowns plus two. Each element for each field variable contains a variant on the field variable. For example, the first element or index contains the value of the field variable at the previous time step. A list of field variable variants is shown in Table 3.2.

The primary variable order and corresponding field variable variant indexing dependents on the operational mode. However, primary variables are always ordered, regardless of the operational mode, according to the following equation sequence: energy, water mass, air mass, oil mass, dissolved-oil mass, salt mass, and surfactant mass. The primary variable ordering system for each operational mode is listed in Table 3.3. Primary variables are referred to by equation in Table 3.3, because primary variables for a given equation and operational mode are dependent on the local phase condition. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of primary variables, phase conditions, and variable switching. Consider an example for the Water-Air Operational Mode, which solves the water mass and air mass conservation equations. For this operational mode there would be four field variable variant indices. Using the aqueous saturation for an unsaturated node as the example field variable, the following variant definitions would apply: the first index would refer to the aqueous saturation at the previous time step, the second index would refer to the aqueous saturation at the current iteration or time step value, the third index would refer to the aqueous saturation with the aqueous pressure incremented, and the fourth index would refer to the aqueous saturation with the gas pressure increments. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of primary variable increments and numerical derivatives.

Flux variables are defined on node surfaces between node points or on boundary surfaces and are represented with two dimensional arrays. Examples of flux variables include heat fluxes, Darcy phase velocities, and component diffusion/dispersion fluxes. Flux variables are aligned with one of the primary orthogonal coordinate directions. Flux variables names which begin with the character "U" are aligned with the x- or radial-direction coordinate, those which begin with the character "V" are aligned with the y- or azimuthal-direction coordinate, and those which begin with the character "W" are aligned with the z-direction coordinate. The first index of a flux variable indicates the increment status and the second index indicates the surface number. Unlike field variables, previous time-step fluxes are not stored. Surface numbers do not correspond directly to node numbers. The number of surfaces in any given coordinate direction equals the number of nodes in that direction plus one.

A list of the flux variable variants is shown in Table 3.4. The primary variable order and corresponding field variable variant index dependents on the operational mode. Refer to Table 3.3 for primary variable ordering sequences as a function of operational mode. In Table 3.4 the upper node refers to the node in the east, north, or top direction (positive x-, y-, or z-direction) with respect to the surface. Likewise the lower node refers to the node in the west, south, or bottom direction (negative x-, y-, or z-direction) with respect to the surface. Consider an example for the Water-Air Operational Mode, which solves the water mass and air mass conservation equations. For this operational mode there would be five flux variable variant indices. Using the z-direction aqueous Darcy velocity for an unsaturated node as the example field variable, the following variant definitions would apply: the first index would refer to the aqueous Darcy velocity for the current iteration or time step value, the second and third indices would refer to the Darcy velocity with the aqueous pressure in the "top" and "bottom" nodes incremented, respectively; and the fourth and fifth indices would refer to the Darcy velocity with the gas pressure in the "top" and "bottom" nodes incremented, respectively. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of primary variable increments and numerical derivatives.

Table 3.1. Glossary of Operational Mode Indices

Index	Operational Mode
1	Water [w/ Transport]
2	Water-Air [w/ Transport]
3	Water-Air-Energy [w/ Transport] [w/ Ice]
4	Water-Oil [w/ Transport]
5	Water-Oil-Air [w/ Transport]
6	Water-Oil-Air-Energy [w/ Transport]
8	Water-Oil-Dissolved Oil [w/ Transport]
9	Water-Oil-Dissolved Oil-Surfactant [w/ Transport]
11	Water-Salt [w/ Transport]
12	Water-Air-Salt [w/ Transport]
13	Water-Air-Energy-Salt [w/ Transport] [w/ Ice]

Table 3.2. Primary Variable Sequencing Scheme

	Primary Variable Sequence			
Index	Equation 1	Equation 2	Equation 3	Equation 4
1	Water			
2	Water	Air		
3	Energy	Water	Air	
4	Water	Oil		
5	Water	Air	Oil	
6	Energy	Water	Air	Oil
8	Water	Oil	Dissolved-Oil	
9	Water	Oil	Dissolved-Oil	Surfactant
11	Water	Salt		
12	Water	Air	Salt	
13	Energy	Water	Air	Salt

Table 3.3. Field Variable Indexing Scheme

Index	Description
1	Previous time step value
2	Current iteration or time step value
3	First primary variable incremented
4	Second primary variable incremented
5	Third primary variable incremented
6	Fourth primary variable incremented

Table 3.4. Flux Variable Indexing Scheme

Index	Description
1	Current iteration or time step value
2	First primary variable in the upper node incremented
3	First primary variable in the lower node incremented
4	Second primary variable in the upper node incremented
5	Second primary variable in the lower node incremented
6	Third primary variable in the upper node incremented
7	Third primary variable in the lower node incremented
8	Fourth primary variable in the upper node incremented
9	Fourth primary variable in the lower node incremented

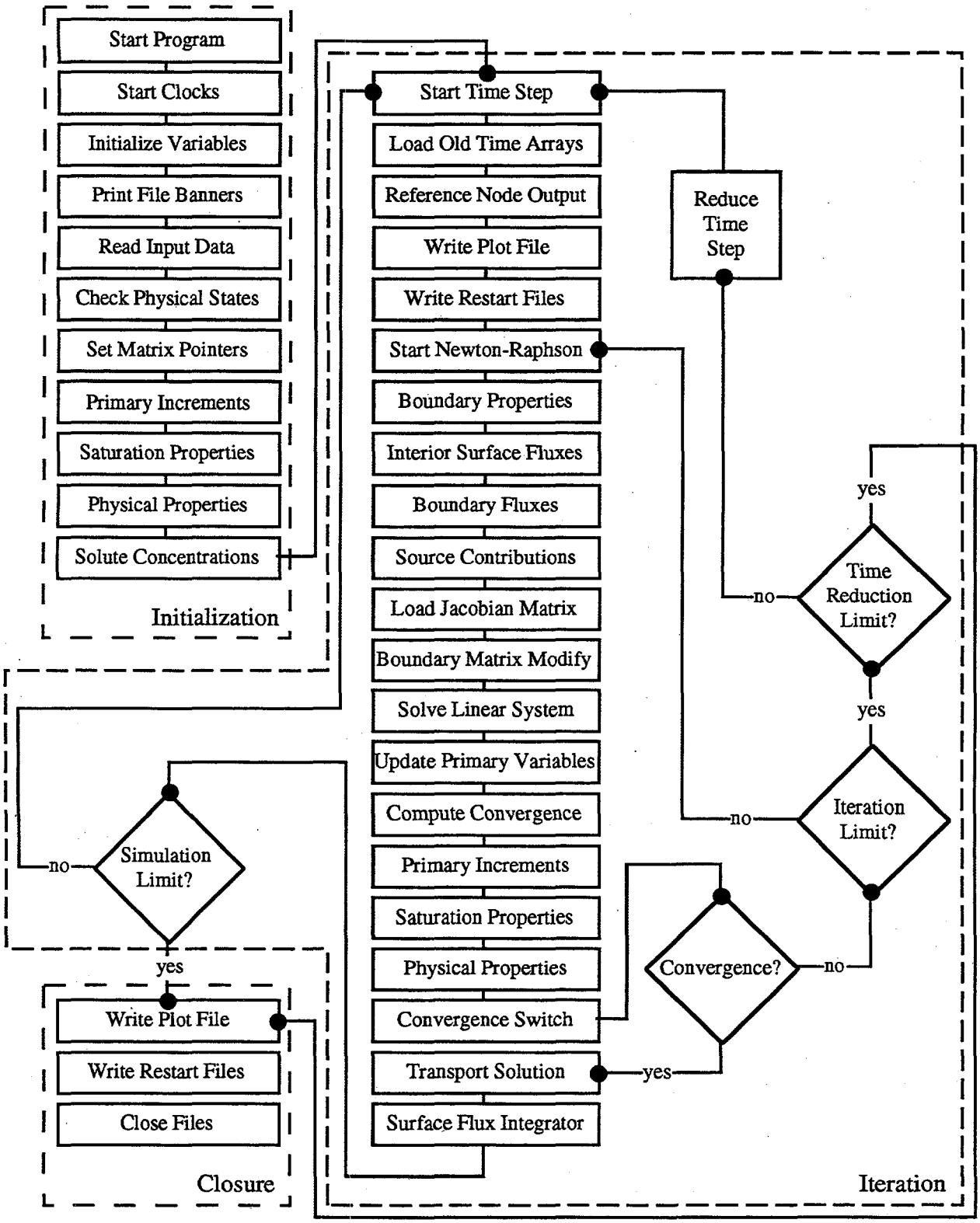

Figure 3.1. STOMP Flow

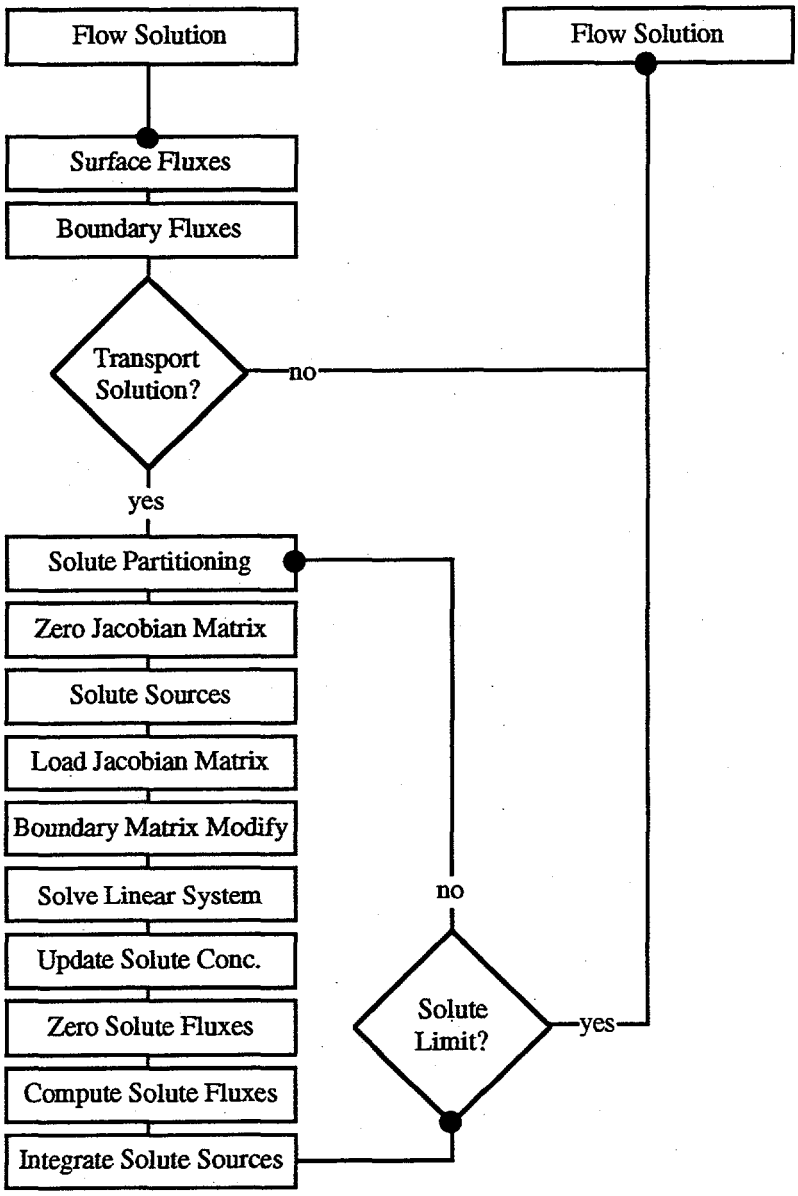

Figure 3.2. Transport Solution Flow Diagram

4.0 Input File

4.1 Introduction

The STOMP simulator is controlled through a text file, which must be entitled "input" for proper execution. This input file has a structured format composed of cards, which contain associated groups of input data. Depending on the operational mode, input cards maybe required, optional, or unused. Required cards must be present in an input file. Optional cards are not strictly required to execute the simulator, but may be required to execute a particular problem. Unused cards are treated as additional text that is unrecognized by the simulator, but will not hinder a proper execution. Cards may appear in any order within the input file; however, the data structure within a card is critical and must follow the formatting directives, shown in the Appendix. Data structures within cards vary with operational mode, which requires the user to follow a series of logic type statements to construct a readable input file. The simulator contains logic to capture and report some input errors. These capabilities are primarily limited to indicating syntax or formatting type errors and will generally not reveal errors such as those associated with ill-posed problems, atypical parameters, or inappropriate grid structures, for example. Because of its text format, STOMP input files may be generated with text editors, word processors, spread sheet programs, or graphical user interfaces and is portable between computing environments. The simulator's read routines are case insensitive (e.g., SAND and sand are equivalent) and allow considerable flexibility in specifying the simulation directives; however, close attention to the formatting instructions in this section will be necessary to prepare an executable input file.

4.2 Input File Structure

A STOMP input file is composed of cards, some of which are required and others which are optional or unused. The number of required cards depends on the operational mode. If an attempt is made to execute the simulator on an "input" file with an incomplete set of required cards, an error message will be generated and the code execution will stop. Optional cards are used to specify STOMP capabilities that may be required to execute a particular problem or generate desired output data. These cards are considered optional, because the capabilities accessed through these cards are not necessarily required to execute the code. Execution of the simulator on input

files with an incomplete set of optional cards yields messages, which will note the missing optional cards but allow the execution to continue. A summary of the required and optional cards which compose a STOMP input file for each operational mode is shown in Table 4.1. Each card begins with a header, which must contain a tilde symbol in the first column followed by the card name (e.g., ~Simulation Title Card. Cards may be arranged in any order within an input file; however, the input format within a card is structured. If a card type is repeated within an input file only the first card will be read by the simulator; the other cards of similar type will be ignored. Blank lines or additional comment lines may be included in the input file outside of the card structures.

4.3 Formatting and Notation

Every input card has formatting specifications, which must be followed to create an input file that is readable by the simulator. The format structure of certain cards varies with the operational mode and/or other input options, and the user must strictly follow the formatting instructions for the particular operational mode of interest. As may be expected, the complexity of the input increases with the number of solved equations. Guides for formatting input cards and card lines are given in the Appendix. The formatting guides shown in the Appendix were written in a compact format using special notations. Definitions of the formatting notation used in Appendix are given in Table 4.2.

Input files are organized into three hierarchical structures, cards, lines, and data. Cards are delimited by a tilde (~) in the first column of the card title. Lines are delimited by hard returns and data are delimited by commas. The simulator recognizes three different types of data, integers, reals and character strings. Integers are used primarily to indicate indices or integral numbers. Integer data must be entered without decimal points or exponential notation. Real data are used to indicate dimensional parameters and can contain decimal points and/or exponential notation. Character string data are primarily used to indicate names, options and units, and are limited to 64 characters unless otherwise noted. Card lines comprise a series of input data items delimited by commas. A comma at the end of the card line is required to close the last data item. The format structure for each line in a card is shown in the Appendix using a format guide (e.g., Format: Integera, Realb, Charc,). Data types in the format statements are indicated as *Integer, Real* and *Char* (i.e., integer, real and character string) with lowercase letter superscripts. The lettered superscripts are used to make correspondence between the data item and it's position in the input line. The total length a single input line cannot exceed 132 characters. Additional text (comments or notes) may appear after the closing comma of an input line. Many input variables contain

default values (indicated with an underline in the Appendix). Default values can be accessed by using a *null* entry. The *null* entry requires that the closing comma for the input data item immediately follow the previous closing comma or only blank spaces fill the space between the closing commas. To start a line with a *null* entry, the closing comma can occur in column 1 or after a number of blank spaces. Example input cards are shown in the Appendix after the formatting instructions for each card. A considerable portion of the input file formatting and creation work can be eliminated by developing input files from previously generated files or through an input generator.

4.4 Units

The simulator offers considerable flexible in specifying units. The user can declare units for both input or output data. Unspecified units will be assumed to be in standard Systeme Internationale (SI) units for the data item. The simulator operates internally almost exclusively in SI units, with pressures expressed in gauge, relative to 1 atm. Unless specifically stated, all input and output pressures are expressed in absolute values. Unit variables are read by the simulator as character strings, translated into primary unit form, and compared against the standard unit form for each data item. During the translation to primary unit form a conversion factor to SI units is generated. A unit character string comprises a combination of the recognized units delimited by spaces and/or a single divisor symbol (i.e., /). Only one divisor may appear in a unit character string. Spaces should not be used to separate the units immediately prior to or following the divisor symbol. The unit strings prior to the divisor symbol are considered as part of the numerator, and conversely the unit strings following the divisor symbol are considered as part of the denominator. Units recognized by the simulator are listed in Table 4.3. The units listed under the subtitle "Miscellaneous Units" are primarily descriptive units which may be included in a unit character string to increase its readability.

The "hc" unit, however, is uniquely reserved for specifying rock/soil hydraulic conductivities, which are normally expressed in velocity units. Without the "hc" unit the input for rock/soil permeability will be interpreted as an intrinsic permeability value. The simulator does not apply hydraulic conductivity values directly, but instead first converts hydraulic conductivity values into intrinsic permeability values using the density and viscosity of water at 20 C. Therefore, when rock/soil permeability values are read as hydraulic conductivities the associated unit character string should include the *"hc"* unit (e.g., hc cm/hr). The unit *"wh"* indicates water equivalent head and can be combined with a length unit for a pressure unit (e.g., wh ft).

As an example, the standard units for thermal conductivity are W/K m, however a user may prefer to specify thermal conductivity in the standard English units of Btu in/hr ft^2 F. In this case the user would enter the character string "Btu in/hr ft^2 F" for the units data item. Note that the carat (i.e., ^) symbol is used to indicate an exponential. During simulator execution, the string of English units for thermal conductivity would be translated into the primary unit form "m/s3 kg K" and compared against the standard unit form. If the primary unit form does not agree with the standard unit form, then an error message is generated by the simulator and the program execution stops. Otherwise if the comparison is successful, then the input data for thermal conductivity, which precedes its unit string, would be converted to SI units.

4.5 Card Descriptions

Formatting instructions for the input cards are provided in the Appendix. This section provides a brief synopsis of each input card with emphasis on its purpose and application. Italicized words refer to specific files, cards, options and data entries shown in the card formats in the Appendix.

4.5.1 Simulation Title Card

This card primarily provides a means to document a simulation. Information recorded in this card is rewritten on the "output" file, which then serves as a permanent record of the simulation. The user is encouraged to use descriptive titles and to briefly record the specifics and purpose of the simulation in the Simulation Notes section of the card. This becomes especially valuable, when the user is making repeated simulations with small modifications to the input parameters. The time required to indicate these changes in either the Simulation Title or Simulation Notes will be invaluable when reviewing archived "output" files.

4.5.2 Solution Control Card

This card controls many general operational aspects of a simulation. Simulations are executed in one of three *Execution Modes*: *Normal*, *Restart*, or *Initial Conditions*. In the *Normal* mode the simulation executes from initial conditions specified with the *Initial Conditions Card*, and

no *"restart"* file is required. The *Restart* mode requires a *"restart"* file generated from a previous execution. Unless specified with an *Overwrite* indicator initial conditions specified on the *Initial Conditions Card* are ignored. Restart files are generated during each *"plot.n"* file write sequence, and have name extensions which correspond to the generating time step (e.g., the file restart.28 would have been generated at the conclusion of time step 28). Restart files contain principal field variables and time information. The *Initial Conditions* mode only executes through the initial checks on the input, no time steps are executed. The *No Flow* option (Water w/ Transport Operational Mode only) is used for transporting solutes through a steady-state flow field. For these simulations the flow field is computed once during the initial time step and then remains unaltered. The *Dynamic Domain* option (Water Operational Mode only) conserves computations by temporarily removing nodes from the computational domain where changes in the flow field are insignificant. The active computational domain is updated every time step.

The *Operation Mode* option determines the governing equations, which will be solved by the simulator. Because execution performance is indirectly proportional to the number of solved governing equations, the user should select an *Operational Mode* which is most appropriate for the problem of interest. For example, if the physical system to be modeled does not contain thermal nor gas pressure gradients and is void of oil, then the most appropriate operational mode would be the Water Operational Mode, where only the water mass conservation equation is solved. Executing the described physical system under the Water-Air Operational Mode would yield nearly identical results, however, the execution time would be significantly increased. Considerable attention should be given to the selection of an appropriate operational mode for the physical system of interest. In terms of efficient and representative simulations, it is equally important not to eliminate a critical physical phenomena through an erroneous assumption, as it is not to solve superfluous governing equations. Each operational mode allows the solution of solute transport equations, which are indicated by including the keyword *Transport* in the operational mode (e.g., Water-Salt w/ Transport or Water-Air and Solute Transport). Solute transport is solved using the Patankar method, unless the keyword *TVD* appears, which indicates the TVD method. The TVD method for coupled salt, dissolved oil or surfactant transport is indicated with the keyword *LFL*.

Execution periods refer to a period of simulation time. The simulator allows the user to specify a single or multiple execution periods. For each execution period the use can control the initial time step, maximum time step, time step acceleration factor, maximum number of Newton-Raphson iterations and convergence criterion. Recommended values for the *Time Step Acceleration Factor*, *Maximum Number of Newton-Raphson Iterations* and *Convergence Criterion* are 1.25, 8, and 1.e-06, respectively. Simulations involving complex phase transitions often

require more Newton-Raphson iterations to reach convergence, because of the design of the phase transition algorithms. For these types of simulations a value of 16 is recommended for the *Maximum Number of Newton-Raphson Iterations*. Except under special circumstances, it is not recommended to change the value for the *Convergence Criterion* from its recommended value. This value has proven through numerous applications to achieve a good balance between accuracy and execution speed.

Field variables, which include physical, thermodynamic, and hydrologic properties are defined in the finite-difference formulation at the node centers. Conversely, flux variables are defined at node interfaces. Computation of flux variables requires knowledge of field variables at node interfaces. Values of flux variables at node interfaces are evaluated by averaging the field values for the two nodes adjoining an interfacial surface. Interfacial averaging schemes may be declared individually for each field variable through the Interfacial Averaging Variables input. The default interfacial averaging schemes for the simulator are shown in Table 4.4. For simulations of physical systems involving heat transfer it should be noted that convergence problems may arise if the density properties are not averaged with upwind weighting. Likewise infiltration problems typically demonstrate strong dependencies on the relative permeability of the infiltrating fluid.

4.5.3 Grid Card

The simulator's finite-difference formulation is based on orthogonal grid systems. Currently two orthogonal grid systems are recognized, the Cartesian and cylindrical coordinate systems. The Cartesian coordinate system is a "right-handed" system with the longitudinal axis (z–direction) aligned with the negative gravitational vector. Cartesian coordinate systems may be defined that are tilted with respect to the gravitational vector. The cylindrical coordinate system has the longitudinal axis (z-direction) aligned with the negative gravitational vector. The radial (r–direction) and azimuthal (q-direction) axes are constrained to a horizontal plane. For the Cartesian coordinate system the terms west, south, and bottom refer to the negative *x-*, *y-*, and *z*-directions, respectively, and the terms east, north, and top refer to the positive *x-*, *y-*, and *z*-directions, respectively. For the cylindrical coordinate system the terms west, south, and bottom refer to the negative *r-*, θ-, and *z*-directions, respectively, and the terms east, north, and top refer to the positive *r-*, θ-, and *z*-directions, respectively. Negative dimensional values are not recognized and axes are defined positive towards increasing node numbers. The grid dimensions which are specified on the *Grid Card* refer to node surfaces; therefore, for grids with nonuniform spacing one

plus the number of nodes entries are required for each grid direction. Node volumes are defined by their bounding surfaces. Cylindrical coordinates systems are restricted to azimuthal axes which are less than or equal to 360 degrees. Refer to the STOMP Theory Guide (White and Oostrom 1996) for graphical descriptions of the Cartesian and cylindrical coordinate systems.

4.5.4 Inactive Nodes Card

This card allows the user to declare nodes, within the computational domain, as inactive or noncomputational. Inactive nodes are those nodes which will remain permanently excluded from the computational domain. Because inactive nodes reduce the number of unknowns, therefore increase execution speed, their use is recommended. Inactive nodes can be used to simulate irregular boundaries or define impervious zones. Inactive nodes can also be used to define internal boundary surfaces. Boundary conditions may be applied to any surface between an active and inactive node. Boundary condition definitions, however, must refer to an active node. Examples of appropriate uses for inactive nodes include representation of the atmosphere above a sloping surface, or conversely a nonhorizontal water table along the bottom of a domain.

4.5.5 Rock/Soil Zonation Card

This card allows the user to partition the computational domain into rock/soil types. All active nodes within the computational domain must have an associated rock/soil type. Rock/soil types are defined with a unique name, which must contain no more than 64 characters. Because most physical systems of interest contain formations or zones of rocks or soils, the computational domain will also contain zones of nodes with hydrogeological similar properties. With this card the user defines the distribution of rock/soil zones, and in subsequent input cards hydrogeologic properties are defined with respect to the defined rock/soil types. This approach prevents the necessity of having to specify hydrogeologic properties uniquely for every node. For simulations using statistically generated rock or soil formations, which require unique hydrogeologic properties for each node, no advantage is gained with this approach. The rock/soil types are extended to the boundary surfaces for nodes adjacent to boundary surfaces. Therefore, field variables for boundary surfaces are computed using the rock/soil type property descriptions from the node adjacent to the boundary surface. Hydrogeologic properties for rock/soil types are specified through the *Mechanical Properties Card, Hydraulic Properties Card, Thermal Properties Card, Saturation Function Card, Aqueous Relative Permeability Function Card, Dissolved-Oil Transport*

Card, Salt Transport Card, Surfactant Transport Card, Gas Relative Permeability Function Card, NAPL Relative Permeability Function Card and *Solute/Porous Media Interaction Card.* A rock/soil type may be repeatedly applied to a node with the last definition being applied. For example, to simplify the zonation of a problem with a dominant rock/soil type and isolated pockets or bands of another rock/soil type, the user should initially declare the dominant rock/soil type as covering the entire problem domain. With subsequent lines, the user can overwrite the original rock/soil type for selected nodes with another rock/soil type. This layering approach is recommended and will often avoid leaving nodes of undeclared rock/soil types.

4.5.6 Mechanical Properties Card

This card allows the user to assign values to the particle density, porosity, specific storativity, compressibility, and tortuosity function for each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. Particle Density represents the rock grain density. This value can be defaulted to 2650 kg/m3 by using a null entry for both the particle density and its associated unit. Total Porosity refers to total connected and unconnected pore volumes. Diffusive Porosity refers to only the connected pore volume. As with Particle Density, the Specific Storativity can be defaulted by using a null entry for both the Specific Storativity and its associated Units. Default specific storativity is computed from the Diffusive Porosity and a default value of 1.x10–71/Pa for the compressibility. Tortuosity functions are required for simulations that involve solute transport or diffusion of components through phases (e.g., water vapor diffusing through the gas phase or dissolved oil diffusing through the aqueous phase). Tortuosities can be computed either as constants, which require input values, or as functions of the phase saturation and diffusive porosity through the Millington and Quirk function. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of the Millington and Quirk tortuosity function.

4.5.7 Hydraulic Properties Card

This card allows the user to assign values to the intrinsic permeability or each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. Intrinsic permeability can be declared directly or through entering the hydraulic conductivity at reference conditions, where reference conditions refer to atmospheric pressure and 20 C. By default the simulator reads the permeability values on this card as intrinsic permeabilities, unless

the character string *hc* is included in the associated units. Default units of m^2 are applied to null entries for the units associated with permeability values. A primary assumption with the simulator is that principal components of the intrinsic permeability tensor are aligned with the principal coordinate directions. For cylindrical coordinate systems the radial, azimuthal, and vertical permeabilities correspond with the *x*-, *y*-, and *z*-direction values, respectively. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of the conversion of hydraulic conductivity at reference conditions to intrinsic permeability.

4.5.8 Thermal Properties Card

This card allows the user to assign values to the thermal conductivity and specific heat for each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. This card is required only for simulations involving the solution of the energy conservation equation. Declaration of the thermal conductivity, depends on the operational mode and function option. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a description of thermal conductivity functions. The Constant option fixes the thermal conductivity to a constant value, independent of temperature or saturation. The Parallel option requires the thermal conductivity of the soil grains and models thermal conductivity with an equivalent parallel path model dependent on porosity, phase saturations, and temperature. The Linear and Somerton options scale the thermal conductivity between the unsaturated and saturated values depending on phase saturation. A primary assumption with the simulator is that principal components of the thermal conductivity tensor are aligned with the principal coordinate directions. For cylindrical coordinate systems the radial, azimuthal, and vertical permeabilities correspond with the x-, y-, and z-direction values, respectively.

4.5.9 Saturation Function Card

This card allows the user to declare and define a saturation-capillary pressure function for each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. Saturation function types and the required input items are primarily dependent on the operational mode. Functional forms for the saturation-capillary pressure functions are preferred, however tabular input is acceptable. By default, tabular data will be interpolated using linear interpolation. Values beyond the table limits will be assigned either the table minimum or maximum values appropriately. For the van Genuchten function, the *"m" Parameter* can be

defaulted (with a *null* entry) or assigned a value. Default values will depend on which porosity distribution model (Mualem or Burdine) is chosen on the relative permeability function cards for the liquid phases. Hysteretic functions are those for which the drainage and imbibition scanning paths differ and include capabilities for entrapment of fluids of lower wettabilities. Fluid entrapment functions consider the hysteresis between wetting and draining paths due only to entrapment of the nonwetting fluids by imbibing wetting fluids. Fluid entrapment functions allow the specification of initial trapped nonwetting fluid saturations without specifying detailed saturation path histories. These functions additionally allow dissolution of the nonwetting phase into the wetting phase (e.g., dissolution of air and oil into the aqueous phase). Nonhysteretic functions require fewer input items than their hysteretic counterparts and use a single scanning path for both drainage and imbibition events. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a complete description of saturation-capillary pressure functions.

4.5.10 Aqueous Relative Permeability Function Card

This card allows the user to declare and define a relative permeability-saturation function for the aqueous phase for each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. Aqueous relative permeability function types and the required input items are dependent on the operational mode. The Mualem and Burdine relative permeability functions are also dependent on the saturation function type and are strictly applicable to the van Genuchten and Brooks and Corey functions. For these functions, either the van Genuchten "m" Parameter or the Brooks and Corey "l" Parameter can be defaulted to the values entered or defaulted with the saturation function. Functional forms for the relative permeability-saturation functions are preferred. Tabular input is, however, acceptable. By default, tabular data will be interpolated using linear interpolation. Values beyond the table limits will be assigned either the table minimum or maximum values appropriately. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a complete description of aqueous relative permeability-saturation functions.

4.5.11 Gas Relative Permeability Function Card

This card is used to declare and define a relative permeability-saturation function for the gas phase for each defined rock/soil type. Every rock/soil type defined on the *Rock/Soil Zonation Card* must be referenced. This card is required only for simulations involving flow and/or

transport through the gas phase. Gas relative permeability function types and the required input items are dependent on the operational mode. The Mualem and Burdine relative permeability functions are also dependent on the saturation function type and are strictly applicable to the van Genuchten and Brooks and Corey functions. For these functions, either the van Genuchten *"m" Parameter* or the Brooks and Corey *"λ" Parameter* can be defaulted to the values entered or defaulted with the saturation function. Functional forms for the relative permeability-saturation functions are preferred, however tabular input is acceptable. By default, tabular data will be interpolated using linear interpolation; cubic spline interpolation is optional. Values beyond the table limits will be assigned either the table minimum or maximum values appropriately. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a complete description of gas relative permeability-saturation functions.

4.5.12 NAPL Relative Permeability Function Card

This card allows the user to declare and define a relative permeability-saturation function for the NAPL for each defined rock/soil type. Every rock/soil type defined on the Rock/Soil Zonation Card must be referenced. This card is required only for simulations involving flow and/or transport through the NAPL phase. The Mualem and Burdine relative permeability functions for the NAPL phase are dependent on the saturation function type and are strictly applicable to the van Genuchten and Brooks and Corey functions. For these functions, either the van Genuchten "m" Parameter or the Brooks and Corey "l" Parameter can be defaulted to the values entered or defaulted with the saturation function. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a complete description of the gas relative permeability-saturation functions.

4.5.13 Oil Properties Card

This card allows the user to declare functions and assign values required to compute the physical properties for an oil or dissolved oil. This card is required only for simulations involving flow and/or transport of an oil or dissolved oil. Properties for oils are computed from critical properties and functional parameters following the Corresponding States Method as presented by Reid et al. (1987). Because the oil property relations within the simulator were primarily based on the formulations in the reference book by Reid et al., the user is encouraged to obtain or borrow a copy of this reference book. Many of the requested input parameters for numerous oils may be

found in the Appendix of this reference. Unless specifically stated below as an optional input, all parameters requested by this card should be considered as required. The *Freezing Point Temperature* is an optional input and can be defaulted with a *null* entry for the parameter and its associated units. The *Boiling Point Temperature* is used in computing oil component diffusion coefficients, oil vapor enthalpy, and liquid oil thermal conductivity; therefore, its value and associated units can be defaulted with *null* entries for simulations which do not involve oil component diffusion nor solution of the energy conservation equation. The *Critical Molar Volume* and its associated units are used only to calculate oil component diffusion coefficients and can be defaulted with *null* entries for simulations not involving oil component diffusion. The *Critical Compressibility* and *Dipole Moment* are used only to calculate the gas viscosity and can be defaulted with *null* entries for simulations which do not solve the gas flow equations. The *Pitzner Acentric Factor* and *Isobaric Molar Specific Heat Constants* are used only to calculate the liquid oil enthalpy and can be defaulted with *null* entries for simulations which do not involve heat transfer nor solution of the energy conservation equation. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a complete description of the Corresponding States Method functions used to compute the oil property functions.

4.5.14 Dissolved-Oil Transport Card

This card allows the user to declare functions and assign values required for dissolved oil transport, dissolved oil transport, or dissolved organic transport. If the operational mode is Water-Oil-Dissolved Oil or Water-Oil-Dissolved Oil-Surfactant, then an Interphase Mass Transfer Function Option must be declared using either the Welty or Parker function and the associated parameters specified. Properties for dissolved oil dispersion and adsorption must be specified with respect to each rock/soil type. For each, the Longitudinal Dispersivity and Transverse Dispersivity of dissolved oil must be specified with respect to each rock/soil type, and a Dissolved-Oil Adsorption Function declared from among the choices of Linear, Linear Kd, Freundlich, or Langmuir functions, with associated parameters for the chosen function specified. Refer to the Appendix for details on the Interphase Mass Transfer Function and Dissolved-Oil Adsorption Function options and required input parameters specified through this card.

4.5.15 Surfactant Property/Transport Card

This card allows the user to define porous media dependent surfactant properties. This card is required only for simulations involving transport of surfactants. For every surfactant defined on the *Solute/Fluid Interaction Card*, every rock/soil type defined on the *Rock/Soil Zonation Card* must be referenced. The *Molecular Weight, Critical Temperature, Critical Pressure, Critical Molar Volume, and Critical Compressibility, Pitzner Accentric Factor, Critical Micellar Concentration, Solubility Function Slope,* and *Oil/Water Interfacial Tension with the Surfactant* are defined for each surfactant. For each surfactant, the porous-media dependent values of the *Longitudinal Dispersivity* (defined with respect to dispersion along the flow path and is assumed to be independent of the flow direction with respect to the porous media structure) and *Transverse Dispersivity* (defined with respect to dispersion transverse to the flow path, independent of the flow direction) must be specified. Other surfactant properties related to the porous media to be defined with this card include the choice for computing the surfactant adsorption. Choices include linear, linear K_d, Freundlich, or Langmuir. Finally, the values for parameters appropriate to the surfactant adsorption choice must be specified. Refer to Section A.15 in the Appendix for additional detail on the input parameters and the equations for surfactant adsorption options.

4.5.16 Salt Transport Card

This card allows the user to define porous media dependent dispersivities for salt. This card is required only for simulations involving salt transport. For operational modes involving salt transport, the salt transport equation is fully coupled with the flow and energy transport equations. For every porous media defined on the *Rock/Soil Zonation Card* dispersivities must be referenced. Computation of effective diffusivity may be either by a conventional or empirical (Kemper and van Schaik) equation. If the empirical equation is selected, a *Molecular Diffusion Rate* must also be specified for each porous media. *Longitudinal Dispersivity* is defined with respect to dispersion along the flow path and is assumed to be independent of the flow direction with respect to the porous media structure. Likewise, *Transverse Dispersivity* is defined with respect to dispersion transverse to the flow path, independent of the flow direction. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a more complete description of the salt mass conservation equation.

4.5.17 Solute/Fluid Interactions Card

This card allows the user to define solutes, solute-fluid interactions, and solute radioactive decay paths. This card is required only for simulations involving transport of solutes. The simulator is capable of simulating any number of solutes with the assumption that solute concentrations remain dilute (solute concentrations do not vary the physical properties of the transporting fluid phases). Solutes can decay radioactively to produce other solutes. For the loose coupling between parent solutes and progeny solutes to function properly, parent solutes must be defined on this card prior to their progeny. The simulator actually solves the transport equation for each solute sequentially in reverse order from the definition list on this card. Solutes are defined by a unique solute name, which must contain no more than 64 characters. Chain Decay Fraction relates a parent decay member with a progeny and refers to the fraction of the decaying parent which produces a particular progeny. Chain Decay Fraction inputs should be fractional values between 0.0 and 1.0, inclusively. Solute-fluid interaction parameters such as molecular diffusion coefficients and interphase partition coefficients depend on the operational mode. Molecular diffusion coefficients refer to the diffusion of the solute through the transporting fluid phase outside of the porous media. Corrections for transport through porous media are handled within the simulator. The interphase partition coefficients define the equilibrium distribution of solute between the active liquid phases. Partition coefficients which define the equilibrium adsorption of solute onto the solid phase are defined on the Solute/Porous Media Interaction Card. The user should carefully note the definitions and requested units for each interphase partition coefficient. The Aqueous-Gas Partition Coefficient is the ratio of the concentration of solute in the aqueous phase per unit mass of aqueous phase to the concentration of solute in the gas phase per unit volume of gas phase. The Aqueous-NAPL Partition Coefficient is the ratio of the concentration of solute in the aqueous phase per unit mass of aqueous phase to the concentration of solute in the NAPL phase per unit mass of NAPL. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a more complete description of the interphase partition coefficients, solute diffusion coefficients, radioactive decay rate equations, and radioactive chain decay fractions.

4.5.18 Solute/Porous Media Interaction Card

This card allows the user to define solid-aqueous phase partition coefficients and porous media dependent hydraulic dispersivities. This card is required only for simulations involving transport of solutes. This card differs from the *Solute/Fluid Interaction Card* because the input parameters declared are dependent on both the solute and rock/soil type. For every solute defined

on the *Solute/Fluid Interaction Card*, every rock/soil type defined on the *Rock/Soil Zonation Card* must be referenced. The *Solid-Aqueous Partition Coefficient* defines the interphase equilibrium of a solute adsorbed on the solid and dissolved in the aqueous phase, and refers to the concentration of solute adsorbed on the solid phase per unit mass of solid phase over the concentration of solute dissolved in the aqueous phase per unit mass aqueous phase. The longitudinal and transverse hydraulic dispersivities are properties only of the rock/soil type. *Longitudinal Dispersivity* is defined with respect to dispersion along the flow path and is assumed to be independent of the flow direction with respect to the porous media structure. Likewise, *Transverse Dispersivity* is defined with respect to dispersion transverse to the flow path, independent of the flow direction. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a more complete description of the solid-aqueous interphase partitioning and hydraulic dispersion of transported solutes.

4.5.19 Initial Conditions Card

This card allows the user to assign starting values to both primary and secondary field variables. The current version of the simulator does not allow the user to initialize flux and/or surface variables. Such initializations would be useful for solving transport problems with steady flow fields. For *restart* simulations, initial conditions are obtained from the *"restart"* file, unless the word *"Overwrite"* appears with the initial condition variable name. Not all initial condition variables can be overwritten. Those variables that can be over written are indicated in the Appendix with the optional text string [Overwrite] show in the *Variable Name Option.* Transport problem simulations with the simulator require the solution of the flow fields. The variables which may be initialized are dependent on the operational mode. For all operational modes, default values for the initial conditions have been specified within STOMP as shown in Table 4.5. For the Water, Water–Air, Water–Air–Energy w/o Ice, Water-Salt, Water-Air-Salt, and Water-Air-Energy-Salt Operational Modes, STOMP allows the user to specify the initial aqueous saturation by assigning initial conditions for any two of the following three variables: gas pressure, aqueous pressure, and aqueous saturation. The unassigned variable will be computed from the other two through the declared saturation-capillary pressure functions. Similarly, for the Water-Air-Energy w/ Ice Operational Mode the user can specify the initial aqueous saturation, frozen water saturation, temperature, and aqueous pressure through combinations on two of the variables, with the remaining two variables being computed by the simulator. No initial saturation options are allowed for the Water-Oil, Water-Oil-Air, or Water-Oil-Air-Energy Operational Modes. For these three-phase operational modes, initial saturations must be declared by specifying initial gas, aqueous, and NAPL pressures. If the initial NAPL pressure is assigned a value below the critical point,

which signifies no NAPL, then the initial NAPL pressure is reset within the simulator to the critical pressure. Therefore, no NAPL conditions can be specified by initializing the NAPL pressure to any value below the aqueous pressure. Initial conditions may be declared repeatedly for a node with the last definition being applied. Initial conditions for solutes are expressed in terms of solute per unit volume where the volume can refer to the total node volume (*Volumetric*), the aqueous-phase volume (*Aqueous Volumetric*), the gas-phase volume (*Gas Volumetric*), or the NAPL volume (*NAPL* Volumetric). Solute units are undefined and can be expressed as the user chooses (e.g., Ci, pCi, gm, kg, mol, kgmol). Although units for expressing solute quantity may vary among solutes, units must be consistent for a single solute among all input data entries.

The gradient utilities of the initial condition card are invaluable and should be used to advantage when possible. Gradients to the initial conditions allow the user to specify that the initial value of a field variables varies along one or more directions in the physical domain. The initial condition variable assigned to an initial condition domain applies to the node with the lowest x-, y-, and z-direction indices. If nonzero gradient values are specified then the initial condition values will vary according to the gradients specified for each physical direction. Default values for the initial condition gradients are zero, indicating no variation. Gradient values are applicable only over the state initial condition domain. An exemplar application of the initial condition gradient utility occurs for problems which start with hydrostatic conditions. For these problems the pressure at the lowest z-direction node positions would be entered with a Z-Dir. Gradient that equaled minus the fluid's density times the acceleration of gravity. For water at 20 C, a z–direction gradient of -9793.5192 1/m will yield constant head conditions, which would be equivalent to hydrostatic conditions. This approach could be used to locate the position of a water table under no-recharge equilibrium conditions given the rock/soil saturation function properties and knowledge of a single pressure using an Initial Condition execution mode simulation.

4.5.20 Boundary Conditions Card

This card allows the user to control the simulation by defining time varying boundary conditions. This card is optional, but is generally necessary to simulate a particular problem. Boundary conditions may be applied to any boundary surface or surface dividing active and inactive nodes. By default all undeclared boundary surfaces have zero flux boundary conditions for both flow and transport. Boundary conditions may be applied only to surfaces of active nodes. To apply a boundary condition to a boundary surface, the surface is referenced by the adjacent active node and a direction with respect to the adjacent node. To apply a boundary condition to a

surface dividing an active and inactive node, the surface is referenced by the active node and the direction to the inactive node with respect to the active node. Boundary conditions are time varying. The user is not allowed to assign multiple boundary conditions to a boundary surface during the same time period, but multiple boundary conditions can be applied to a boundary surface over different time periods. The simulator controls time steps to agree with time transitions in boundary conditions.

Application of boundary conditions requires an appropriate conceptualization of the physical problem and translation of that conceptualization into boundary condition form. The variety of boundary condition types available in the simulator should afford the user with the flexibility to solve most subsurface flow and transport problems. The boundary condition card reader within the simulator performs limited error checking on the boundary condition inputs. An error free boundary condition card does not guarantee the user has not created an ill-posed problem or an execution which will successfully converge. For example, a mistake frequently made by users is to specify infiltration rates at the top of a column with positive fluxes. While this input would be perfectly acceptable to the boundary condition input reader, the specified condition would actually withdraw flux from the top of the column since the z-axis and z-direction flux are positive in the upward direction.

The *Boundary Surface Direction* is specified with respect to the active node adjacent to a boundary surface. For the Cartesian coordinate system the terms west, south, and bottom refer to the negative x-, y-, and z-directions, respectively, and the terms east, north, and top refer to the positive x-, y-, and z-directions, respectively. For the cylindrical coordinate system the terms west, south, and bottom refer to the negative r-, θ-, and z-directions, respectively, and the terms east, north, and top refer to the positive r-, θ-, and z-directions, respectively.

Boundary condition inputs depend on the operational mode, with the required inputs becoming more complex with increasing number of solved equations. Generally, the number of boundary type which must be declared for each boundary equals the number of solved governing flow and transport equations. The *Dirichlet* boundary type is used to specify a field value (e.g., pressure, temperature, or solute concentration) at the boundary surface. The *Neumann* boundary type allows the user to specify a flux (e.g., liquid phase flux, heat flux, or solute flux) at the boundary surface. The *Zero Flux* boundary type is used to impose no flow and/or transport conditions across the boundary. The *Saturated* boundary type is available only for two-phase conditions and imposes total-liquid saturation conditions (e.g., water table) at the boundary surface. The *Unit Gradient* boundary type imposes hydrostatic conditions across the boundary

surface for the specified phase. The *Hydraulic Gradient* boundary type should be applied only to a column or plane of vertical surfaces. With this boundary type the user specifies a fluid phase pressure at the lowest surfaces of a column or row and the simulator then computes fluid phase pressure for the remaining boundary surfaces assuming hydrostatic conditions for the fluid phase. The *Seepage* Face boundary type is similar to a Hydraulic Gradient boundary, but is limited to pressure boundaries of the local gas pressure. This boundary type is designed to model an exposed vertical face that "seeps" liquids. Liquid can enter a seepage face only for phase pressures that exceed the local gas pressure. The Initial Conditions boundary type fixes the boundary field variables (e.g., pressure, temperature, or solute concentration) to the initial value of the field variables of the node adjacent to the boundary surface. This boundary type is invariant with time. Inflow and Outflow boundary types are applicable only to solute and energy boundary conditions. These boundary types consider only advectively transported solute or energy; diffusion transport across the boundary surface is neglected. The solute concentration boundary types (e.g., Volumetric Concentration, Aqueous Conc., Gas Conc., and NAPL Conc.) are equivalent to Dirichlet boundary types for solute transport. These boundary types differ by their definitions of solute concentration.

Time variations of the boundary conditions are controlled through declaring multiple boundary times. All Boundary Time inputs are referenced against the Initial Time specified on the Solution Control card or obtained from a "restart" file. A boundary condition declared with a single Boundary Time implies that the boundary condition is time invariant and the specified Boundary Time represents the start time for the boundary condition. Prior to the start time the boundary surface will be assumed to be of type Zero Flux. The specified boundary condition will remain in effect from the start time until the execution completion. If a boundary condition is declared with multiple Boundary Times, then the first time listed equals the start time, the last time listed equals the stop time, and the intermediate times are transition points. For simulation times outside of the start and stop time limits, *Zero Flux* boundary conditions apply. For simulation times between two *Boundary Times*, linear interpolation of the boundary conditions is applied. Step boundary condition changes can be simulated by defining duplicate *Boundary Times* the first time would indicate the completion of the previous boundary condition and the second time would indicate the start of the new boundary condition. At the completion of the step boundary condition another set of duplicate *Boundary Time* declarations would be used. Step boundary conditions are convenient methods for introducing slugs of fluids, heat, or solute in conjunction with the *Neumann* boundary type.

Regardless of the boundary type, the boundary condition inputs are used to compute phase saturations, phase relative permeabilities, and physical properties at the boundary surfaces. For the more complex operational modes, especially those modes involving three-phase conditions, it is critical for the user to specify boundary conditions which yield appropriate secondary field variables at the boundary surface. For example, a three-phase system with infiltrating NAPL under ponded conditions could be declared as type *Zero Flux* for the aqueous phase, *Zero Flux* for the gas phase, and *Dirichlet* for the NAPL. For this boundary condition scenario, the aqueous and gas pressures requested for the *Zero Flux* type boundary are used with the specified NAPL pressure to compute liquid saturation conditions at the boundary surface. If the pressures specified yield zero NAPL saturation, then the NAPL relative permeability would equal zero and no NAPL would infiltrate across the surface, in spite of the ponding conditions.

4.5.21 Source Card

This card allows the user to control sources and/or sinks of mass, energy, and solutes by defining timing varying sources. By definition sinks are negative sources, and sources refer to an influx of mass, energy, or solute into a node. Sources can be specified for interior or boundary nodes and are functionally analogous to Neumann type boundary conditions. Sources applied to inactive nodes are not recognized. Sources are time varying; however, unlike boundary conditions multiple sources may be applied to a node during the same time period. The simulator controls time steps to agree with time transitions in sources. Source inputs depend on the operational mode, with the required inputs becoming more complex with increasing number of solved equations. Sinks withdraw mass, energy, or solutes from a node. The physical properties for the fluids withdrawn through sinks equal those of the node. Sources inject mass, energy, or solutes into a node. The physical properties for fluids injected through sources are computed from the specified input parameters.

Sources of type Aqueous Volumetric and Aqueous Mass inject aqueous fluid. If specified, dissolved air and oil will be injected with the aqueous fluid. For nonisothermal problems the injected fluid would also transport an amount of heat into the node based on the enthalpy of the entering aqueous fluid. Sources of type *Gas Volumetric w/ Mass Fraction*, *Gas Volumetric w/ Relative Humidity*, *Gas Mass w/ Mass Fraction*, and *Gas Mass w/ Relative Humidity* are similar to the aqueous sources, in that mass and heat injected into the node would include contributions from water vapor, air, and oil vapors. The source type suffix *w/ Mass Fraction* indicates that water vapor, air, and vapor concentrations in the gas phase will be specified through inputs of mass

fractions for these quantities. Similarly, the source type suffix *w/ Relative Humidity* indicates that water vapor, air, and oil vapor concentrations in the gas phase will be specified through inputs of relative humidities for the water and oil. Sources of type *NAPL Volumetric* and *NAPL Mass* inject NAPL. Because dissolution of air and water in the NAPL phase is neglected by the simulator, then these sources represent pure NAPL. Energy sources are specified as type *Power* or *Power Density*, where *Power* type sources inject energy per unit time and *Power Density* type sources inject energy per unit time per node volume. *Solute* sources inject solute mass in the assumed units for solute (e.g., Ci, pCi, gm, kg, mol, kgmol).

Well sources model production or injection wells using a nonlinear well model. Positive flow rates are considered as injection wells and negative flow rates are considered as production wells. The well model essentially computes the depth of liquid (water) in the well through a Newton-Raphson iteration scheme that balances flux across the well casing with the source pumping rate. For injection wells, the depth of water in the well is limited by a well bottom pressure constraint, expressed in pressure units. For pumping wells the contribution from seepage flow is ignored. Wells are vertical features in the STOMP simulator and must be defined over a contiguous array of vertical nodes. By default well bore storage is considered in the well model and the well volume, as defined by the *Borehole Radius* is removed from the node volume. Both of these features can be ignored by including the key words *No Storage* and *No Volume* in the *Source Type Option.*

Time variations of sources are controlled through declaring multiple source times. All *Source Time* inputs are referenced against the *Initial Time* specified on the *Solution Control* card or obtained from a *"restart"* file. A source declared with a single *Source Time* implies that the source is time invariant and the specified *Source Time* represents the start time for the source. Prior to the start time the source will be zero, and from the start time to execution completion the source will be as specified. If a source is declared with multiple *Source Times*, then the first time listed equals the start time, the last time listed equals the stop time, and the intermediate times are transition points. For simulation times outside of the start and stop time limits, zero source conditions apply. For simulation times between two *Source Times*, linear interpolation of the sources is applied. Step source changes can be simulated by defining duplicate *Source Times* the first time would indicate the completion of the previous source and the second time would indicate the start of the new source. At the completion of the step source another set of duplicate *Source Time* declarations would be used. Step sources are convenient methods for introduction slugs of fluids, heat, or solute into an interior node.

4.5.22 Output Control Card

This card allows the user to control output written to the "output" file, "plot" file, and "screen" (i.e., standard input/output device (STDIO)). The "output" file contains an interpreted and reformatted version of the "input" and simulation results for selected variables at selected "reference nodes" over the simulation period. The "plot.n" file contains values of geometric parameters and selected variables for the entire computational domain (both active and inactive nodes) at selected simulation times. A "plot.n" file will always be generated at the conclusion of an execution. The output to the STDIO primary comprises the reference node variable results versus simulation time and/or time step. It is recommended that the user request "screen" output, because well chosen output maybe invaluable in tracking the simulation progress and identifying possible input errors. If a suite of repetitive simulations are being performed, then "screen" output can be reduced to minimum values. As with other input cards, output options are dependent on the operational mode. With respect to this card, output options primarily refer to computed field and flux variables.

Reference node output is generated by selecting reference nodes and output variables. The user may request any number of reference nodes, but reference node output was primarily designed for tracking the time evolution of selected variables at key nodes of interest. Reference nodes are defined with three indices, which indicate the *x*-, *y*-, and *z*-direction coordinates of the node. Node numbering in the simulator increments in the order i, j, and k, where the indices refer to the *x*-, *y*-, and *z*-directions, respectively, for Cartesian coordinate systems and *r*-, θ-, and *z*-directions, respectively, for cylindrical coordinate systems. *Reference Node Screen Output Frequency* and *Reference Node Output File Frequency* are parameters which indicate how often with respect to time step reference node output will be written to STDIO and the *"output"* file, respectively. A frequency value of 1 indicates reference node output occurs every time step, whereas a frequency value of 10 indicates reference node output occurs every 10 time steps. The user has control over the output time and length units and the number of significant digits reported to the various output media. Unless declared through the *Output Time Units* or *Output Length Units* input items, values for time and lengths recorded to the output media will be expressed in units of seconds and meters, respectively. These inputs allow the user customize time- and length-scale units to those most appropriate to the solved problem. *The Screen Significant Digits*, *Output File Significant Digits*, and *Plot File Significant Digits* input items allow the user to customize the number of significant digits which appear in field and flux variable results written to the STDIO, *"output"* file, and *"plot.n"* file, respectively. Default values for the number of significant digits is 5, and the minimum number of significant digits is 4. Field and flux variables output for the

reference nodes are selected from the list shown for each operational mode in the Appendix. The same list of variables for each operational is available for output to the *"plot.n"* files. Output units for all variables with units can be specified immediately following the variable name. Variables without units require a *null* entry for the variable units. *null* entries for variables with units yield default output units, which are expressed in SI units.

Plot files are written at the conclusion of an execution, by default, and at each requested *Plot File Output Time*. Plot files contain geometry data and selected field and flux variable results for every node in the computational domain. These files represent a "snapshot" of the simulation at a certain point in time. Requests for *Plot File Output Times* can be specified with user defined units. A *"restart.n"* file is generated with every *"plot.n"* file. Both *"restart.n"* and *"plot.n"* files are suffixed with an file name extension of a dot followed by an integer (e.g., plot.567, restart.32). The extension integer corresponds with the time step for which the file was written. Both *"plot.n"* and *"restart.n"* files are written at the conclusion of a time step. Field and flux variables recorded to *"plot.n"* files are selected from the list shown for each operational mode in the Appendix (refer to the reference node variable options). Output units for all variables with units can be specified in the input item immediately following the variable name. Variables without units require a null entry for the variable units. Null entries for variables with units yield default output units, which are expressed in SI units.

4.5.23 Surface Flux Card

This card allows the user to define surfaces to track fluxes of fluid mass, fluid volume, heat, and/or solutes. A surface defined with this card can be composed of rectangular areas of coplanar surfaces on exterior boundaries or between interior nodes. Output from the surface flux integration routines are written to the *"surface"* file and contain flux rate and integral data for each defined surface at every time step. The types of fluxes which can be tracked depend on the operational mode. Declaration of surfaces is similar to defining boundary condition surfaces. Surfaces are defined by referencing a group of coplanar nodes and a surface direction with respect to the nodes. For the Cartesian coordinate system the terms west, south, and bottom refer to the negative x-, y-, and z-directions, respectively, and the terms east, north, and top refer to the positive x-, y-, and z-directions, respectively. For the cylindrical coordinate system the terms west, south, and bottom refer to the negative r-, θ-, and z-directions, respectively, and the terms east, north, and top refer to the positive r-, θ-, and z-directions, respectively. For example, a surface to track the flux rate and integral of a particular solute entering the water table could be

defined for a simulation with saturated conditions along the bottom boundary surface by referencing the node group along the bottom of the computational grid and defining the *Surface Flux Orientation* as *Bottom.* One surface flux rate and integral value is computed for each defined surface and represents the summation of surface flux contributions from the individual surfaces in the coplanar group of surfaces.

Table 4.1. Required and Optional Input Cards

Operational Mode	Card Name	Card Status
All	Inactive Nodes	Optional
	Initial Conditions	Optional
	Boundary Conditions	Optional
	Source	Optional
	Output Control	Optional
	Surface Flux	Optional
	Solute/Fluid Interactions	Optional
	Solute/Porous Media Interactions	Optional

Operational Mode	Card Name	Card Status
Water	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required

Operational Mode	Card Name	Card Status
Water-Air	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	Gas Relative Permeability Function	Required

Operational Mode	Card Name	Card Status
Water-Air-Energy	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Thermal Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	Gas Relative Permeability Function	Required

Table 4.1. Cont'd

Operational Mode	Card Name	Card Status
Water-Oil	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	NAPL Relative Permeability Function	Required
	Oil Properties	Required

Operational Mode	Card Name	Card Status
Water-Oil-Dissolved Oil	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	NAPL Relative Permeability Function	Required
	Oil Properties	Required
	Dissolved Oil Transport	Required

Operational Mode	Card Name	Card Status
Water-Oil-Dissolved Oil-Surfactant	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	NAPL Relative Permeability Function	Required
	Oil Properties	Required
	Dissolved Oil Transport	Required
	Surfactant Properties/Transport	Required

Table 4.1. Cont'd

Operational Mode	Card Name	Card Status
Water-Oil-Air	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	NAPL Relative Permeability Function	Required
	Gas Relative Permeability Function	Required
	Oil Properties	Required

Operational Mode	Card Name	Card Status
Water-Oil-Air-Energy	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Thermal Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	NAPL Relative Permeability Function	Required
	Gas Relative Permeability Function	Required
	Oil Properties	Required

Operational Mode	Card Name	Card Status
Water-Salt	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	Salt Transport	Required

Operational Mode	Card Name	Card Status
Water-Air-Salt	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	Gas Relative Permeability Function	Required
	Salt Transport	Required

Table 4.1. Cont'd

Operational Mode	Card Name	Card Status
Water-Air-Energy-Salt	Simulation Title	Required
	Solution Control	Required
	Grid	Required
	Rock/Soil Zonation	Required
	Mechanical Properties	Required
	Hydraulic Properties	Required
	Thermal Properties	Required
	Saturation Function	Required
	Aqueous Relative Permeability Function	Required
	Gas Relative Permeability Function	Required
	Salt Transport	Required

Table 4.2. Input Format Notation Guide

Notation	Description
{ Option }	Character string options are indicated by enclosing braces. Options are chosen by entering word(s) within the braces, exactly as shown. Only one option should be chosen for each data entry.
[Optional]	Enclosing brackets indicate optional characters or words. These characters can be entered to include the input file to improve its readability or to specify optional features.
{{ Contains }}	Indicates the option contains a particular word. For example "Fractured Tuff" contains the word "Fractured" thus indicating a dual-porosity type rock/soil.
< *Data Types* >	Indicates repeated formatting.
Char[a]	Character string data type, referenced by superscript "a".
Integer[a]	Integer data type (no character data or decimal points) reference by superscript "a".
Real[a]	Real data type (decimal points and exponential notation are acceptable), reference by superscript "a".
#	A pound symbol in the first column indicates a comment line and will be ignored during execution. Comment lines may be placed inside or outside card structures. All lines outside of the card structures are ignored during execution.
~ Card Name	A tilde symbol in the first column indicates the start of a new card.
,	Data entries are comma delimited. Commas shown in the line format structures must be entered as shown, including a closing comma at the end of each line. Characters following the last comma of a data line are ignored during execution.
Units[a] (m)	Indicates the SI unit for the input data item referenced by superscript "a".
Format:	Indicates line formatting instructions and the beginning of a new input line. Each format statement requires a new input line.

Table 4.2. Cont'd

Notation	Description
Endcard:	Indicates end of a card.
For: Integer Instructions **Endfor:** Integer	Indicates instruction looping.
If: Name: Card = { Opt_1 } Instructions1 **Elseif:** Name: Card = { Opt_2 } Instructions2 **Elseif:** Instructions3 **Endif:**	Indicates decision logic.
IfDef: Opt_1 Instructions1 **ElseifDef:** Opt_2 Instructions2 **ElseDef:** Instructions3 **EndifDef:**	Indicates C preprocessor options and logic.
Note:	Indicates formatting information.

Table 4.3. Recognized Units

Notation	Description	SI Equivalent	Conversion to SI	Base Units
1	one	N/A	N/A	N/A
aqu	aqueous phase	N/A	N/A	N/A
aqueous	aqueous phase	N/A	N/A	N/A
atm	atmosphere	Pa	101325.0	$M / L T^2$
bar	bar	Pa	$1. \times 10^5$	$M / L T^2$
btu	BTU	J	1054.4	$M L^2 / T^2$
c	Celcius	C	1.	K
cal	calorie	J	4.184	$M L^2 / T^2$
ci	Curie	N/A	N/A	N/A
cm	centimeter	m	$1. \times 10^{-2}$	M / L T
cp	centipoise	Pa s	$1. \times 10^{-3}$	M / L T
d	day	s	1/86400.	T
darcy	darcy	m^2	9.8697×10^{-13}	L^2
day	day	s	1/86400.	T
debyes	Debyes	N/A	N/A	$(M/L T^2)^{1/2}$
deg	degrees	rad	0.017453	N/A
degrees	degrees	rad	0.017453	N/A
dyn	dynes	N	$1. \times 10^{-5}$	$M L / T^2$
dynes	dynes	N	$1. \times 10^{-5}$	$M L / T^2$
f	Farhenheit	C	(f-32) x (5/9)	K
ft	foot	m	0.3048	L
furlong	furlong	m	2.01168×10^2	L
g	gm	kg	$1. \times 10^{-3}$	M
gal	gallon	m^3	3.7854×10^{-3}	L^3
gas	gas phase	N/A	N/A	N/A
gm	gm	kg	$1. \times 10^{-3}$	M
gram	gm	kg	$1. \times 10^{-3}$	M
hc	hydraulic cond.	N/A	1.03910×10^{-7}	N/A
hour	hour	s	1/3600.	T
hp	horsepower	W	745.7	$M L^2 / T^3$
hr	hour	s	1/3600.	T
in	inch	m	2.54×10^{-2}	L
j	Joule	J	1.	$M L^2 / T^2$
k	kelvin	C	1.	K
kg	kilogram	kg	1.	M
kgmol	kilogram-mole	kgmol	1.	Mo
l	liter	m3	1. x 10-3	L^3

Table 4.3. Cont'd

Notation	Description	SI Equivalent	Conversion to SI	Base Units
lb	pound	kg	0.45359	M
lbm	pound	kg	0.45359	M
lbmol	pound-mol	kgmol	0.45359	Mo
liq	liquid	N/A	N/A	N/A
liter	liter	m^3	$1. \times 10^{-3}$	L^3
m	meter	m	1.	L
mg	milligram	kg	1.0×10^{-6}	M
min	minute	s	1/60.	T
ml	milliliter	m^3	1.0×10^{-6}	L^3
mm	millimeter	m	1.0×10^{-3}	L
mol	mole	kgmol	1.0×10^{-3}	Mo
mole	mole	kgmol	1.0×10^{-3}	Mo
napl	NAPL	N/A	N/A	N/A
oil	oil	N/A	N/A	N/A
p	Poise	Pa s	1.0×10^{-1}	M / L T
pa	Pascal	Pa	1.0	M / L T^2
pci	picoCuries	N/A	N/A	N/A
psi	lbs/in^2	Pa	6894.8	M / L T^2
r	Rakine	K		K
rad	radians	rad	1.	N/A
rod	rod	m	5.0292	L
s	second	s	1.	T
sec	second	s	1.	T
slug	slug	kg	14.594	M
sol	solid phase	N/A	N/A	N/A
solid	solid phase	N/A	N/A	N/A
voc	VOC phase	N/A	N/A	N/A
w	Watt	W	1.	M L^2 / T^3
water	water	N/A	N/A	N/A
week	week	s	1/604800.	T
wh	water head	Pa	9.79353×10^3	M / L T^2
wk	week	s	1/604800.	T
yd	yard	m	0.9144	L
year	year	s	1/31557600.	T
yr	year	s	1/31557600.	T

Table 4.4. Default Interfacial Averaging Options

Field Variable	Interfacial Averaging
Aqueous Density	Upwind
Aqueous Relative Permeability	Upwind
Aqueous Viscosity	Harmonic
Dissolved Air Diffusion	Harmonic
Dissolved Oil Diffusion	Harmonic
Dissolved Salt Diffusion	Harmonic
Gas Density	Upwind
Gas Relative Permeability	Upwind
Gas Viscosity	Harmonic
Hydraulic Dispersion	Harmonic
Intrinsic Permeability	Harmonic
NAPL Density	Upwind
NAPL Relative Permeability	Upwind
NAPL Viscosity	Harmonic
Oil Vapor Diffusion	Harmonic
Solute Diffusion	Harmonic
Thermal Conductivity	Harmonic
Water Vapor Diffusion	Harmonic

Table 4.5. STOMP Initial Conditions Default Values

Initial Field Variable	Symbol	Default Value
temperature	T	20.D+0
pressure of phase j	P_j	101325.D+0
saturation of phase j	s_j	0.D+0
mole fraction of component i in phase j	χ_j^i	0.D+0
solute concentration in phase γ	C_γ	0.D+0
salt concentration in phase γ	S_γ	0.D+0

5.0 Parameters File

5.1 Introduction

This section describes the format and contents of the "parameters" file, which is required to compile and execute the STOMP simulator. Parameters are used by the FORTRAN programing language and compilers to allocate memory for storage of variables. The FORTRAN language is unable to allocate memory dynamically, therefore all memory storage requirements must be defined at compliation time. No execution errors will occur if the memory allocated is greater that required by the simulation, unless the memory requirements exceed the computer's capabilities. Unless necessary, the user should avoid executing simulations which require the use of virtual memory. The time required to swap data between the virtual memory storage device and the active memory typically yields poor execution speeds. The STOMP simulator requires two types of parameters (declared and computed) to be defined, prior to compilation. The user is responsible for properly assigning all of the declared parameters. Declared parameters are assigned by modifying the "parameters" file supplied with the STOMP simlator using a text editor (word processor) or by creating a new "parameters" file. The equations for the computed parameters must be included in each "parameters" file after the declared parameters. The parameter definitions given in this manual represent minimum acceptable values. All declared parameters, except for "switch" type parameters, must have minimum values of 1. Undersized parameters will generally yield execution errors, which may or may not be detected by the system. Oversized parameters are permissible, but can result in excessive memory allocation.

5.2 Solution/Simulation Parameters

The "LNOTES" parameter equals the number of lines in the "Simulation Notes" field and should equal the variable "Number of Simulation Note Lines" on the Simulation Title Card. The "LEPD" parameter equals the number of execution periods and should equal the variable "Number of Execution Periods" on the Solution Control Card. A simulation with 7 lines of simulation notes and 4 execution periods would have a solution/simulation parameter set that appeared as

```
PARAMETER(LNOTES=7, LEPD=4)
```

5.3 Computational Domain Parameters

The "LFX," "LFY," and "LFZ" parameters equal the number of nodes in the x-, y- and z-directions, respectively, for Cartesian coordinate systems or the number of nodes in the radial-, -azimuthal, and z-directions, respectively, for cylindrical coordinate systems. These parameters should equal the variables "X-Dir. Nodes," "Y-Dir. Nodes," and "Z-Dir. Nodes" on the Grid Card. The parameter "LAN" equals the number of active nodes a should equal the total number of nodes minus the number of inactive nodes. The parameter "LAD" equals the number of active dimensions, where active dimensions are coordinate directions with more than 1 node. The number of active dimensions should be a number between 1 and 3, inclusive. The parameter "LMNP" equals the minimun plane size and should equal the minimun of the following three products: LFX*LFY, LFY*LFZ, and LFZ*LFX. A simulation with two active dimensions, having 22, 1, and 103 nodes in the x-, y-, and z- directions, respectively, and 51 inactive nodes would have a computational domain parameter set that appeared as

```
PARAMETER(LFX=22, LFY=1, LFZ=103)
PARAMETER(LAN=2215, LAD=2, LMNP=22)
```

5.4 Operation Mode Switch Parameters

The parameters "LT," "LL," "LG," "LN," and "LC" are switches for the energy, water mass, air mass, VOC mass, and solute conservation equations. For these "switch" type parameters, a value of 1 indicates the equation is solved and a value of 0 indicates the equation is unsolved. The parameter "LFC" is a switch for freezing conditions. A value of 0 indicates nonfreezing conditions, whereas a value of 1 indicates freezing conditions. The parameter "LS" is a switch for coupled salt- or surfactant-mass transport. The parameter "LD" is a switch for the dissolved-oil mass conservation equation. The switch combinations set with these parameters should agree with the operational mode. A simulation using the Water-Air w/ Transport Operational Mode would have a operational mode switch parameter set that appeared as

```
PARAMETER(LT=0, LL=1, LG=1, LN=0, LC=1, LFC=0, LS=0, LD=0)
```

5.5 Linear System Solver Switch Parameters

The parameters "LBD" and "LCG" are switches for the banded and conjugate gradient linear system solvers. For these "switch" type parameters, a value of 1 indicates the solver is active and and value of 0 indicates the solver is inactive. Only one solver can be active for a simulation, either the banded matrix or the conjugate gradient solver. These parameters switches are the only means the user has for declaring a linear equation solution scheme. A simulation that used the banded matrix linear system solver would have a linear system solver parameter set that appeared as

```
PARAMETER(LBD=1, LCG=0)
```

5.6 Boundary Condition Parameters

The parameters "LBC" and "LBTM" equal the number of boundary surfaces and maximum number of boundary condition times. A "Boundary Condition Domain" as declared on the Boundary Conditions Card may contain more than one boundary surface. The parameter "LBC" indicates the total number of boundary surfaces, not the number of boundary condition domains. Although they may refer to the same computational domain surface, boundary surfaces declared in one boundary condition domain differ from those declared in another. The maximum number of boundary condition times equals the maximun number of times declared for all boundary condition domains. A simulation with three boundary condition domains, the first with 6 boundary surfaces and 13 boundary condition times, the second with 2 boundary condition surfaces and 26 boundary condition times, and the third with 34 boundary condition surfaces and 2 boundary condition times would have a boundary parameter set that appeared as

```
PARAMETER(LBC=42, LBTM=26)
```

5.7 Source Parameters

The parameters "LSR" and "LSTM" equal the number of source domains and maximum number of source times. Unlike the boundary condition parameters, a single source may contain more than one node, but is only counted as once with respect to the parameter "LSR." The maximum number of source times equals the maximum number of times declared for all sources.

A simulation with three source domains, the first with 6 nodes and 13 source times, the second with 2 sources and 26 source times, and the third with 34 sources and 2 source times would have a source parameter set that appeared as

```
PARAMETER(LSR=3, LSTM=26)
```

5.8 Rock Type and Solute Transport Parameters

The parameters "LRC" and "LSOLU" equal the number of rock types and number of transported solutes. The number of rock types should equal the number of rock/soil types listed on the Rock/Soil Zonation Card. Because one rock/soil type can define the rock/soil type for more than one rock/soil zonation domain the number of rock/soil types does not equal the "Number of Rock/Soil Zonation Domains" declared on the Rock/Soil Zonation Card. Each rock/soil type should have a unique name. The number of solutes should equal the number of solutes defined on the Solute/Fluid Interactions Card. Each solute should have a unique name. A simulation with 14 rock/soil zonation domains, 3 rock/soil types, 7 solutes for the Water-Air w/ Transport Operational Mode would have a rock types and solutes parameter set that appeared as

```
PARAMETER(LRC=3, LSOLU=7)
```

5.9 Output Parameters

The parameters "LREF," "LPTM," and "LSF" equal the number of reference nodes, number of "plot" file output times, and the number of flux surfaces, respectively. The number of reference nodes should equal the "Number of Reference Nodes" input on the Output Control Card. The number of "plot" file output times should equal the "Number of Plot File Times" input on the Output Control Card. The "plot.n" file which is generated, by default, at the conclusion of an execution should not be counted as a "plot" file time. The number of flux surfaces should equal the "Number of Surface Flux Inputs" input on the Surface Flux Card. A simulation with 4 reference nodes, 6 "plot" file output times (other than the "plot" file generated at the conclusion of the execution), and 23 flux surfaces would have an output parameter set that appeared as

```
PARAMETER(LREF=4, LPTM=6, LSF=23)
```

5.10 *k-S-P* Function Parameters

The parameters "LTBL", "LPATH" and "LCHEM" equal the total number of table entries and the number of scanning paths for hysteretic saturation functions, and the number of chemical reactions, respectively. The number of table entries should equal the sum of "Number of Table Entries" on the Saturation Function, Aqueous Relative Permeability Function, and Gas Relative Permeability Function Cards. Tabular NAPL relative permeability functions are not recognized by the STOMP simulator. The number of scanning paths should equal the "Number of Scanning Paths" input for hysteretic saturation functions on the Saturation Function Card. The recommend number of scanning paths for a hysteretic saturation function is 7. For nonhysteretic saturation functions the number of scanning paths should equal 1. A Water Operational Mode simulation with 2 rock/soil types, 34 tabular entries for the saturation function for the first rock/soil type, a declared nonhysteretic saturation function for the second rock/soil type, 27 tabular entries for the aqueous relative permeability for the first rock/soil type, and 31 tabular entries for the aqueous relative permeability for the second rock/soil type would have a k-S-P function parameter set that appeared as

```
PARAMETER(LTBL=92, LPATH=7, LCHEM=0)
```

5.11 Computed Parameters

Computed parameters are arithmetic combinations of the declared parameters and must follow the declared parameters in the *"parameters"* file. Computed parameters related to the conjugate gradient linear system solver are declared within the source code. Definitions of the computed parameters in the "parameters" file are given in Table 5.1. The computed parameter set for all simulations appears as

```
PARAMETER(LUK=LT+LL+LG+LN+LS+LD, LPH=LL+LG+LN, LCMP=LL+LS+LD)
PARAMETER(LFXY=LFX*LFY, LFYZ=LFY*LFZ, LFZX=LFZ*LFX)
PARAMETER(LFD=LFX*LFY*LFZ)
PARAMETER(LNE=(LUK*LUK*(7*LFD-2*LFXY-2*LFYZ-2*LFZX))**LUM)
PARAMETER(LHBW=LUK*LMNP+LUK-1)
PARAMETER(LJA=LBD + LCG*LAN*LUK + LUM*LAN*LUK)
PARAMETER(LJB=(2*LAN*LUK)**LUM, LJC=LAN**LUM)
PARAMETER(LJD=LBD*(3*LHBW+1) + LCG*LAN*LUK + LUM*6*LNE)
PARAMETER(LJE=LBD*LAN*LUK + LCG*((2*LAD+1)*LUK+2*LAD) + LUM)
PARAMETER(LJF=LAN*LUK)
PARAMETER(LJG=LBD*(3*LHBW+1) + LCG*LAN*LUK + LUM)
PARAMETER(LJH=LBD*LAN*LUK + LCG*(2*LAD+1) + LUM)
PARAMETER(LJJ=LBD*LAN*LUK + LCG + LUM)
```

```
PARAMETER(LSV=LUK+2, LSFV=2*LUK+1)
PARAMETER(LSX=(LFX+1)*LFY*LFZ)
PARAMETER(LSY=LFX*(LFY+1)*LFZ)
PARAMETER(LSZ=LFX*LFY*(LFZ+1))
PARAMETER(LFDT=LFD**LT,LFDL=LFD**LL,LFDG=LFD**LG,LFDN=LFD**LN)
PARAMETER(LFDC=LFD**LC, LFDI=LFD**LFC, LFDS=LFD**LS, LFDD=LFD**LD)
PARAMETER(LSXT=LSX**LT,LSXL=LSX**LL,LSXG=LSX**LG,LSXN=LSX**LN)
PARAMETER(LSXC=LSX**LC, LSXS=LSX**LS, LSXD=LSX**LD)
PARAMETER(LSYT=LSY**LT,LSYL=LSY**LL,LSYG=LSY**LG,LSYN=LSY**LN)
PARAMETER(LSYC=LSY**LC, LSYS=LSY**LS, LSYD=LSY**LD)
PARAMETER(LSZT=LSZ**LT,LSZL=LSZ**LL,LSZG=LSZ**LG,LSZN=LSZ**LN)
PARAMETER(LSZC=LSZ**LC, LSZS=LSZ**LS, LSZD=LSZ**LD)
PARAMETER(LRCT=LRC**LT,LRCL=LRC**LL,LRCG=LRC**LG,LRCN=LRC**LN)
PARAMETER(LRCC=LRC**LC, LRCI=LRC**LFC, LRCS=LRC**LS, LRCD=LRC**LD)
PARAMETER(LBCT=LBC**LT,LBCL=LBC**LL,LBCG=LBC**LG,LBCN=LBC**LN)
PARAMETER(LBCC=LBC**LC, LBCI=LBC**LFC, LBCS=LBC**LS, LBCD=LBC**LD)
PARAMETER(LBCU=LUK+LPH+LT+2, LBCV=LBCU+LSOLU)
PARAMETER(LOUPV=200+11*(LSOLU))
PARAMETER(LJI=LBD*LAN*LUK + LCG + (3*LNE+23*LFD*LUK+9)*LUM)
PARAMETER(LSCHR=18)
```

Table 5.1. Computed Parameter Definitions

Parameter	Definition
LUK	Number of solved coupled equations
LPH	Number of phases
LCMP	Number of components
LFXY	Number of nodes in the xy plane
LFYZ	Number of nodes in the yz plane
LFZX	Number of nodes in the zx plane
LFD	Number of nodes
LHBW	Jacobian matrix half-band width
LJA	Conjugate gradient solver parameter
LJD	Jacobian matrix coefficient array parameter
LJE	Jacobian matrix coefficient array parameter
LJF	Solution vector parameter
LJG	Linear system solver parameter
LJH	Linear system solver parameter
LJJ	Linear system solver parameter
LSV	Number of field variable indices
LSFV	Number of flux variable indices
LSX	Number of nodes in the x direction
LSY	Number of nodes in the y direction
LSZ	Number of nodes in the z direction
LFDT	Number of energy field variables
LFDL	Number of water field variables
LFDG	Number of air field variables
LFDN	Number of oil field variables
LFDC	Number of solute field variables
LFDI	Number of ice field variables
LFDS	Number of salt/surfactant field variables
LFDD	Number of dissolved oil field variables
LSXT	Number of energy flux variables in the x direction
LSXL	Number of water flux variables in the x direction
LSXG	Number of air flux variables in the x direction
LSXN	Number of oil flux variables in the x direction
LSXC	Number of solute flux variables in the x direction
LSXS	Number of salt/surfactant flux variables in the x direction
LSXD	Number of dissolved oil flux variables in the x direction
LSXT	Number of energy flux variables in the y direction
LSXL	Number of water flux variables in the y direction
LSXG	Number of air flux variables in the y direction
LSXN	Number of oil flux variables in the y direction
LSXC	Number of solute flux variables in the y direction
LSXS	Number of salt/surfactant flux variables in the y direction
LSXD	Number of dissolved oil flux variables in the y direction
LSXT	Number of energy flux variables in the z direction
LSXL	Number of water flux variables in the z direction
LSXG	Number of air flux variables in the z direction
LSXN	Number of oil flux variables in the z direction
LSXC	Number of solute flux variables in the z direction
LSXS	Number of salt/surfactant flux variables in the z direction
LSXD	Number of dissolved oil flux variables in the z direction

Table 5.1. Cont'd

Parameter	Definition
LRCT	Number of energy rock/soil variables
LRCL	Number of water rock/soil variables
LRCG	Number of air rock/soil variables
LRCN	Number of oil rock/soil variables
LRCC	Number of solute rock/soil variables
LRCI	Number of ice rock/soil variables
LRCS	Number of salt/surfactant rock/soil variables
LRCD	Number of dissolved rock/soil variables
LBCT	Number of energy boundary condition variables
LBCL	Number of water boundary condition variables
LBCG	Number of air boundary condition variables
LBCN	Number of oil boundary condition variables
LBCC	Number of solute boundary condition variables
LBCI	Number of ice boundary condition variables
LBCS	Number of salt/surfactant boundary condition variables
LBCD	Number of dissolved boundary condition variables
LBCU	Number of coupled boundary condition variables
LBCV	Number of total boundary condition variables
LOUPV	Number of output variable options
LSCHR	Number of k-s-P variable indices

6.0 Compilation and Execution

6.1 Introduction

The STOMP simulator is written in the FORTRAN 77 language, following American National Standards Institute (ANSI 1978) standards, and was designed primarily for execution on computers with UNIX[a] operating systems. An assembled source code, however, can be compiled and executed on any computer with an ANSI FORTRAN compiler, provided that the machine has sufficient memory. Although there is a general correspondence in computer design between memory size and execution speed, sufficient memory to compile and execute is not an assurance that the machine will have sufficient execution speeds to complete a simulation within a reasonable time period. The simulator should be thought of as a collection of source code libraries. The libraries which become part of the executable code dependent on the operational mode. Each operational mode requires a particular group of library source codes. The normal procedure for building an executable version for a particular operational mode involves compiling the required group of library source codes, using the declared and computed parameters in the "parameters" file to define memory requirements, and linking the compiled object files to create an executable. An alternate option would be to assemble or concatenate all of the source code libraries required for a particular operational mode into a single file and then subsequently compile the assembled source code. Either procedure can be performed manually or automatically. The "make" utility (Talbott 1988), which generates a sequence of commands for execution by the UNIX shell, can be used to automatically compile and link the required source code libraries for a particular operational mode. A "makefile," which contains a "make" utility instruction set has been built to generate an executable or assembly source code for the simulator.

6.2 Compilation

Four assembly options are recognized for the simulator, "source," "standard," "debug," and "optimize." The "source" option will concatenate all of the source coding required for an operational mode into a single source , which can be compiled. The "source" option generates a

---

[a] UNIX is a registered trademark of AT&T Information Systems.

file with a ".f" name extension. The "standard" option compiles and links all of the source and object coding required for an operational mode into a single executable file. The "standard" option generates a file with a ".e" name extension. The "debug" option is similar to the "standard" option except that a symbol table was generated for the executable, which allows code debugging through UNIX utilities such as "dbx," "xdb," or "dbxtool." The "debug" option generates a file with a ".d" name extension. The "optimize" option is similar to the "standard" option except that the code is compiled with the default level of optimization. The "optimize" option generates a file with a ".x" name extension. Assembly options are entered on the UNIX shell command line with the call to "make." Operational mode options are specified on the command line through integer indices. Table 6.1 shows the correspondence between operational modes and their indices. As an example, the following UNIX shell command string on a computer which supports the "make" utility would generate an optimized executable of the Water-Air-Energy Operational Mode

```
make optimize mode=3
```

where, a file named "stomp3_bd.x" would be generated. The "_bd" suffix on the file name refers to the banded matrix linear system solver, which is the default. To generate a debug executable for the Water-Oil Operational Mode that invoked the conjugate gradient solver the following UNIX shell command would be used

```
make debug mode=4 -e solver=cg
```

where, a file named "stomp4_bd.d" would be generated and the "`-e solver=cg`" option on the command line is used to override the default value for the "solver" macro definition of "bd" within the "makefile." To generate the source coding for the Water Operational Mode with chemical reactions the following UNIX shell command would be used

```
make source mode=1 -e decay=reactive
```

where, a file name "stomp1_bd.f" would be generated and the "`-e decay=reactive`" option on the command line is used to override the default value for the solute decay scheme of "radioactive."

The assembly process has certain requirements with respect to source files, the *"commons"* file, the *"parameters"* file, and the "make" description file. The recommended directory structure for assembling the simulator is to store all of the source files (those files which have a ".f" or ".F" extensions) and the *"commons"* file in a single directory. A path name to this directory may be

defaulted into the "make" description file or provided on the UNIX shell command line. For example to generate a standard executable for the Water Operational Mode, with the source files stored in the directory "/home/user/stomp/source," the following UNIX shell command would be used

```
make standard mode=1 -e DIR=/home/user/stomp/source
```

where, the "`-e DIR=/home/user/stomp/source`" macro directs the compiler and C preprocessor to search the "/home/user/stomp/source" directory for the source code libraries and include files. The *"parameters"* file should be reside in the current directory. The "make" description file is normally named either *"makefile"* or *"Makefile"* and should reside in the current directory. However, the "make" utility allows the user to specify a description file with a nonstandard name, using the -f option (Talbott 1988). The *"parameters"* file should reside in the current directory. Four source files (band.F, clocks.F, intlz.F, and pnspcg.F) have ".F" extensions, which serve to indicate that these files will be preprocessed with the C preprocessor "cpp." The C preprocessor interprets the machine type and linear system solver macros and generates the appropriate FORTRAN coding.

Executing "make" will generate several files in the current directory depending on the assembly option. Source code assembly will convert required files with ".F" extensions through the C preprocessor to FORTRAN source code files with ".f" extensions and produce a single concatenated source code file with a ".f" extension. Executable assembly will also convert the required files with ".F" extensions through the C preprocessor and produce an executable file. The executable assembly, however, will also produce object files, with ".o" extensions for every required FORTRAN source file. These object files will appear in the current directory but are not necessary for code execution and can be discarded. Any modifications to the *"parameters"* file would require that the code be recompiled and linked. Executing "make" following a modification to the parameter file will automatically regenerate the object files and executable. Because of the updating features in "make," any recompilation using a different assembly option will require the user to "touch" the *"parameters"* file or remove all of the object and FORTRAN source files from the current directory. For example, if a user wanted to generate both a standard and debug version of a Water Operational Mode executable the following command sequence would be required

```
make standard mode=1
...
touch parameters
...
make debug mode = 1
```

where, the "touch parameters" command simply applies a new time and date stamp to the *"parameters"* file. The "make" utility simply automates the assembly process. The same steps could be performed manually.

6.3 Execution

Executing the simulator is a straight forward and only requires that the executable version of the code and an input file named "input" reside in the current directory. For restart simulations a restart file named "restart" must also reside in the current directory. Because restart files are created with an extension that corresponds with the generating time step, the user must rename the appropriate restart file to "restart". For a UNIX operating system, execution is started by typing in the name of the executable file. Execution will be indicated by the printing of a STOMP title banner and program disclaimer to the standard input/output device (e.g., screen). Two types of error messages may be generated, during an STOMP execution. The first type is a system generated message that typically indicates a memory, FORTRAN, or other system error identified by the system. The second type of error messages are those generated by the STOMP code, which typically refer to input, parameter, or convergence failure type messages. STOMP generated messages are divided into three categories according to severity. The most severe are "ERROR" messages, which abort the program execution. Undersized parameters are typical of errors which yield "ERROR" messages, because execution of the simulator with undersized parameters may yield gross errors, or even worse subtle errors which may pass undetected in the results. Next on the severity level are the "WARNING" messages, which generally warrants notice by the user that a problem with the input file probably exists. The least severe are "NOTE" messages, which are used to record events like the absence of an optional input card.

When an execution is terminated with a STOMP generated "ERROR" message two types of information appear on the standard input/output device. The first piece of information is an error message with or without related data that indicates the nature of the error. Input formatting errors are trapped to a limited degree by the simulator. Unfortunately, the input error message that results may not always agree with the actual location of the identified input error. This is because the original error may have skipped detection. For example, a missing input item on an input line would not be identified until STOMP identified an incorrect data type or the end of the line was reached. The user should expect to find input errors slightly before the location indicated by the generating error message. The second type of information, which is generated with an "ERROR"

message is a calling sequence. The calling sequence is a slash delineated list of routine names which were called prior to the identified error. The calling sequence for an input error on the Aqueous Relative Permeability Card for the Water Operational Mode would appears as

```
/STOMP1/RDINPT1/RDRPL/WRMSG
```

where the called routine names are delineated with a forward slash. For this example, the main routine "STOMP1" called the subroutine "RDINPT1" to read the input file for the Water Operational Mode, which called "RDRPL" the global routine to read the Aqueous Relative Permeability Card, which identified an error and called "WRMSG" to write an error message to the standard input/output device (e.g., screen).

Table 6.1. Glossary of Operational Mode Indices

Index	Operational Mode
1	Water [w/ Transport]
2	Water-Air [w/ Transport]
3	Water-Air-Energy [w/ Transport] [w/ Ice]
4	Water-Oil [w/ Transport]
5	Water-Oil-Air [w/ Transport]
6	Water-Oil-Air-Energy [w/ Transport]
8	Water-Oil-Dissolved Oil [w/ Transport]
9	Water-Oil-Dissolved Oil-Surfactant [w/ Transport]
11	Water-Salt [w/ Transport]
12	Water-Air-Salt [w/ Transport]
13	Water-Air-Energy-Salt [w/ Transport] [w/ Ice]

7.0 Output Files

7.1 Introduction

The STOMP simulator can generate, depending on the requested output and saturation function type, two data files for restarting a simulation and three simulation result files. Every execution produces an output file named *"output"*. If the simulation concludes without a fatal error, then a *"restart.n"* and a *"plot.n"* file are also generated. Restart and plot file names include extensions (i.e., *".n"*) which correspond to the time step for which the file was written. For example, a restart file named "restart.39" would have been written at the conclusion of the 39th time step. Simulations involving hysteretic saturation functions also produce a *"hyster.n"* file, which contains saturation history records. As with the restart and plot files, hyster file names also include time-step extension. A surface file named *"surface"* is generated whenever a Surface Flux Card is included in the *"input"* file. All output files generated by the simulator (*"output"*, *"restart.n"*, *"plot.n"*, *"hyster.n"* and *"surface"* files) are ASCII text files.

7.2 Output File

An output file named *"output"* is generated with every execution of the simulator. If a file named *"output"* already exists in the current directory, then that file will be erased and overwritten. The user is responsible for renaming previous *"output"* files prior to executing the simulator. An *"output"* file begins with a welcome statement, disclaimer, and banner, which should appear similar to

```
Welcome to ...

                              STOMP
        Subsurface Transport Over Multiple Phases


This file was produced by STOMP, a numerical simulator
developed by the Pacific Northwest National Laboratory, with
support from the VOC-Arid Integrated Demonstration Project,
Office of Technology Development, U.S. Department of Energy.
Results from this version of STOMP should not be used for
license related applications.

For inquiries or assistance:  Call (509) 372-6070

                         ---  OUTPUT  ---
```

Following this banner is an input record, which documents the simulation input. An input record is included on the *"output"* file to document the simulation in the event that the *"input"* file is destroyed or becomes separated from the *"output"* file. The input record is formatted differently that an *"input"* file, but contains all of the information listed on an *"input"* file. Optional cards which are not included in the *"input"* file are noted in the input record. If an input error is identified by the simulator, an error message will appear in the input record at the point the error was noted. Input cards are read by the simulator in a specific order and will appear in the input record in that order. Because input cards can be organized randomly on the *"input"* file the card order on the input record may not match that on the *"input"* file. The reference node output record follows the input record.

The reference node output record is a table of simulation data and selected reference node variables, which are printed according the frequency requested on the *Output Control Card* with the input item *Reference Node Output File Frequency*. A table column header is printed to delineate the columns every 10 print records. Definitions of the header abbreviations for the reference node output are given in Table 7.1. Each print record will show simulation data and reference node

variable data for each reference node requested. Simulation data comprises the time step, the reference node number, the simulation time, the simulation time step, and the number of Newton-Raphson iterations required to reach convergence. Reference node variable data comprise values of the variables specified on the *Output Control Card* through the *Reference Node Variable* inputs. Reference nodes are indexed by node number. The corresponding *x*-, *y*-, and *z*-direction indices are shown for each reference node on the line preceding the column headers. For example, a simulation with reference node 272 at *x*-, *y*-, and *z*-direction indices of 2, 3, and 6, respectively would have a reference node header line that appeared as

```
Reference Node(s) (  2, 3,  6:     272)
```

Column headers for reference node variables are delineated with an abbreviated title and associated units. Units are enclosed in brackets below the variable abbreviation. A portion of reference node output record for a horizontal heat pipe problem involving 50 nodes for time steps 70 through 73 appeared as

```
Reference Node(s) (  1,  1,  1:     1) ( 50,  1,  1:    50)
 Step  Node     Time      Timestep  Itr     T           SL          PL
              [day   ]   [day   ]        [c     ]                 [pa     ]
   70     1  2.837E+01  1.793E+00   8  7.1893E+01  9.9893E-01  1.0133E+05
   70    50  2.837E+01  1.793E+00   8  1.3564E+02  2.0992E-02 -1.9896E+08
   71     1  3.061E+01  2.242E+00   9  7.1895E+01  9.9903E-01  1.0133E+05
   71    50  3.061E+01  2.242E+00   9  1.4039E+02  1.9265E-02 -2.2722E+08
   72     1  3.341E+01  2.802E+00   9  7.1878E+01  9.9933E-01  1.0133E+05
   72    50  3.341E+01  2.802E+00   9  1.4598E+02  1.7489E-02 -2.6050E+08
   73     1  3.692E+01  3.503E+00   8  7.1869E+01  9.9951E-01  1.0133E+05
   73    50  3.692E+01  3.503E+00   8  1.5225E+02  1.5759E-02 -2.9758E+08
```

where, the reference nodes are nodes 1 and 50, and the reference node variable data includes the temperature, reported in degrees Celsius, the aqueous saturation, and the aqueous pressure, reported in Pascal (absolute).

Convergence failures and subsequent time step reductions are also noted within the reference node output record. Refer to Section 6.4 for a discussion on convergence errors. Three types of convergence error messages may appear within the output. The first type indicates that the update to a primary variable exceeded the maximum allowable change. An example of this type of convergence error for a horizontal heat pipe problem appeared as

```
      ---  Excessive Primary Variable Change  ---
Temperature =  3.80318E+02 Node =     50
Water Pressure = -1.33730E+09 Node =     50
Gas Pressure =  3.07342E+05 Node =     50
Time Step Reduced From  2.9383E+00 day to  5.8766E-01 day
```

where, an excessive change to at least one primary variable was noted at node 50 and the time step was reduced from 2.938 to 0.5877 days and repeated. The second type indicates that a converged solution was not obtained within the maximum number of Newton-Raphson iterations. An example of this type of convergence error for a three-phase volatile infiltration problem appeared as

```
      ---  Convergence Failure  ---
Water Equation Maximum Residual =  1.3313E-02 Node =    152
Air Equation Maximum Residual =  8.2280E-07 Node =     76
VOC Equation Maximum Residual =  1.0472E-02 Node =    154
Time Step Reduced From  7.0121E-03 hr to  1.4024E-03 hr
```

where, the maximum normalized residuals for the water and oil mass conservation equations were noted, which exceeded the convergence criterion. After this convergence failure the time step was reduced from 0.00701 to 0.00140 hours and repeated. The third type indicates that the linear system solver has failed to reach a solution. An example of this type of convergence error for a three-phase volatile infiltration problem appeared as

```
      ---  Singular Matrix  ---
Jacobian Matrix Index =    339
Node Number           =    113
Time Step Reduced From  1.4849E-02 hr to  2.9697E-03 hr
```

where, a zero or nearly zero diagonal was noted for the oil equation at node 113. After this convergence failure the time step was reduced from 0.0148 to 0.00297 hours and repeated. All three types of error messages include additional information that pertains to the specific convergence problem. Convergence errors result in a reduction in the time step with a repeated attempt to solve the system of governing equations. Four successive convergence failures without an intermediate successfully converged time steps results in termination of the execution. Output files are concluded with the following closing message

```
      ---  End of STOMP Simulation  ---
```

7.3 Plot File

A plot file is generated by default at the conclusion of every execution and otherwise when requested on the Output Control Card with the Plot File Output Times input items. If a file named "plot.n" (where n refers to the time step extension) already exists in the current directory then the file will be erased and overwritten. The user is responsible for renaming previous "plot.n" files prior to executing the simulator. A plot file begins with a welcome statement, disclaimer, and banner, which should appear similar to

```
Welcome to ...

                          STOMP
      Subsurface Transport Over Multiple Phases

This file was produced by STOMP, a numerical simulator
developed by the Pacific Northwest National Laboratory, with
support from the VOC-Arid Integrated Demonstration Project,
Office of Technology Development, U.S. Department of Energy.
Results from this version of STOMP should not be used for
license related applications.

For inquiries or assistance:  Call (509) 372-6070

                    ---  PLOT  ---
```

Following this header are data on the current time step, simulation time, and number of nodes in three coordinate directions. The remaining portion of a *"plot.n"* file comprises arrays of geometric and variable data. These data are arranged in groups, each with a title line that indicates the type of data which follows. Data groups are arranged in rows with 10 data items per row. Each group of data contains a field or surface variable listed sequentially for every node or surface in the computational domain. The first four groups of data appear in every *"plot.n"* file and contain the x- or radial-direction node positions, y- or azimuthal-direction node positions, z-direction node positions, and node volumes for every node in the computational domain. The remaining groups of data correspond to the requested *Plot File Variables*. Field variable data groups list the value of the field variable at the node centroid. Surface variables and/or fluxes are written to the plot file twice. The first flux variable data group lists the value of the flux variable at the node centroid, whereas the second group lists the value at the node surfaces. Node centroid values for flux variables are simple arithmetic averages of the two node surface values.

7.4 Surface File

A surface file named *"surface"* is generated during an execution if surfaces were defined on the *Surface Flux Card.* If a file named *"surface"* already exists in the current directory then that file will be erased and overwritten. The user is responsible for renaming previous *"surface"* files prior to executing the simulator. A *"surface"* file begins with a welcome statement, disclaimer, and banner, which should appear similar to

```
Welcome to ...

                          STOMP
       Subsurface Transport Over Multiple Phases

This file was produced by STOMP, a numerical simulator
developed by the Pacific Northwest National Laboratory, with
support from the VOC-Arid Integrated Demonstration Project,
Office of Technology Development, U.S. Department of Energy.
Results from this version of STOMP should not be used for
license related applications.

For inquiries or assistance:  Call (509) 372-6070

                      ---  SURFACE  ---
```

Following the header are columns of surface flux rate and integral data. Surface flux information is written to the *"surface"* file every time step, at the conclusion of the time step. The first column of data in the *"surface"* file is the time step in units, which are specified with the *Output Time Units* variable on the *Output Control Card.* Each surface flux output is written in two columns, the first column being the surface flux rate at the current time step and the second being the integral of the surface flux rate from the beginning of the current simulation. Surface flux integral data are not carried over on restart simulations. Each column in the *"surface"* file is titled with an abbreviated header and associated units shown in brackets below the column title. Definitions of the header abbreviations for the *"surface"* file are given in Table 7.2. Each header title is followed with an integer enclosed in parentheses. The integer refers to the surface flux number, as defined on the *Surface Flux Card* (i.e., the surface flux definition order).

7.5 Restart File

A restart file is generated by default at the conclusion of every execution and otherwise when requested on the *Output Control Card* with the *Plot File Output Times* input items. If a file named *"restart.n"* (where *n* refers to the time step extension) already exists in the current directory then the file will be erased and overwritten. The user is responsible for renaming previous *"restart.n"* files prior to executing the simulator. A restart file begins with a welcome statement, disclaimer, and banner, which should appear similar to

```
Welcome to ...

                          STOMP
      Subsurface Transport Over Multiple Phases

This file was produced by STOMP, a numerical simulator
developed by the Pacific Northwest National Laboratory, with
support from the VOC-Arid Integrated Demonstration Project,
Office of Technology Development, U.S. Department of Energy.
Results from this version of STOMP should not be used for
license related applications.

For inquiries or assistance:  Call (509) 372-6070

                    ---  RESTART  ---
```

All data values in restart files are in SI units with pressures expressed in gauge. The first data line of a restart file contains the following timing and simulation information: time, time step, maximum time step, time step acceleration factor, convergence criterion, maximum number of Newton-Raphson iterations, number of time steps, and number of nodes. The remaining lines are node data lines, which contain the following field variable data for the current time step for every node (one line per node): temperature, aqueous pressure, gas pressure, NAPL pressure, oil vapor pressure, water vapor pressure, aqueous saturation, gas saturation, and NAPL saturation. If the simulation includes solute transport then the node data lines will also contain solute volumetric concentrations for each solute. Moreover, if the simulation includes freezing water conditions then the node data lines will also contain the frozen water pressure and frozen water saturation. All node data lines are terminated with an integer value for the phase condition. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a discussion of phase conditions.

7.6 Hyster File

Hyster files are generated only for simulations with hysteretic saturation functions. A hyster file is generated by default at the conclusion of every execution and otherwise when requested on the *Output Control Card* with the *Plot File Output Times* input items. If a file named *"hyster.n"* (where *n* refers to the time step extension) already exists in the current directory then that file will be erased and overwritten. The user is responsible for renaming previous *"hyster.n"* files prior to executing the simulator. A hyster file begins with a welcome statement, disclaimer, and banner, which should appear similar to

```
Welcome to ...

                              STOMP
        Subsurface Transport Over Multiple Phases

 This file was produced by STOMP, a numerical simulator
 developed by the Pacific Northwest National Laboratory, with
 support from the VOC-Arid Integrated Demonstration Project,
 Office of Technology Development, U.S. Department of Energy.
 Results from this version of STOMP should not be used for
 license related applications.

 For inquiries or assistance:  Call (509) 372-6070

                        ---  HYSTER  ---
```

All data values in hyster files are in SI units. The first data line of a hyster file contains the following simulation and timing information: time, time step, number of time steps, and number of nodes. The next group of data contains the following integer index data for every node (one line per node): phase condition number, water saturation path number, total liquid saturation path number, maximum water saturation path-type index, maximum total liquid saturation path-type index, water saturation path-type index, total-liquid saturation path-type index. Refer to the STOMP Theory Guide (White and Oostrom 1996) for a discussion of phase conditions. Even saturation path numbers indicate wetting paths and odd saturation path numbers indicate drying paths. A saturation path-type index equals 0 for drying paths and 1 for wetting paths. The next group of data contains pressure head and saturation history data for every node (one line per node) in the following sequence: previous time step scaled capillary head for water saturation, previous time step scaled capillary head for total-liquid saturation, previous time step effective NAPL saturation, effective air residual saturation (air-water systems), effective air residual saturation (air-oil systems), effective oil residual saturation (oil-water systems), apparent water saturation at the reversal from main drainage (air-water systems), historic minimum apparent water saturation for

trapped NAPL, historic minimum apparent water saturation for free NAPL, and the historic apparent total saturation. The next line of data contains an integer value for the number of saturation scanning paths. The next four groups of data are values of saturation and capillary head at the scanning path reversal points. The data are arranged by variable, node, and reversal points. For each variable, one line of reversal point data is written for each node. The reversal point variables are written in the following sequence: apparent water saturation at the reversal points, apparent total-liquid saturation at the reversal points, capillary head for water saturation at the reversal points, and capillary head for total-liquid saturation at the reversal points.

Table 7.1. Reference Node Output Record Abbreviations

Abbreviation	Definition
AST	Apparent total liquid saturation
ASW	Apparent aqueous saturation
Cn[a]	Solute volumetric concentration
CGn[a]	Solute gas volumetric concentration
CLn[a]	Solute aqueous volumetric concentration
CNn[a]	Solute NAPL volumetric concentration
CNGA	Air mass concentration in the gas
CNGO	VOC mass concentration in the gas
CNGW	Water mass concentration in the gas
CNLA	Air mass concentration in the aqueous
CNLO	VOC mass concentration in the aqueous
CNLW	Water mass concentration in the aqueous
CS	Salt volumetric concentration
CSL	Salt aqueous volumetric concentration
DSGF	Dual porosity gas saturation (fracture)
DSGM	Dual porosity gas saturation (matrix)
DSLF	Dual porosity aqueous saturation (fracture)
DSLM	Dual porosity aqueous saturation (matrix)
GPG	Gas pressure (gauge)
GPL	Aqueous pressure (gauge)
GPN	NAPL pressure (gauge)
HHG	Gas hydraulic head (water equivalent)
HHL	Aqueous hydraulic head (water equivalent)
HHN	NAPL hydraulic head (water equivalent)
MCL	Aqueous liquid content
MCN	NAPL content
MCT	Total liquid content
PFW	Ice pressure (absolute)
PG	Gas pressure (absolute)
PHCN	Phase condition index
PL	Aqueous pressure (absolute)
PN	NAPL pressure (absolute)
PORD	Diffusive porosity
RHOF	Ice density

[a] n refers to a solute number

Table 7.1. Cont'd

Abbreviation	Definition
RHOG	Gas density
RPL	Aqueous relative permeability
RPN	NAPL relative permeability
RSZN	Rock/Solute Zone Number
SFW	Ice saturation
SG	Gas saturation
SL	Aqueous saturation
SN	NAPL saturation
ST	Total-liquid saturation
SRn[(a)]	Solute source integral
SRCA	Air mass source integral
SRCO	VOC mass source integral
SRCT	Energy source integral
SRCW	Water mass source integral
ST	Total liquid saturation
T	Temperature
TA	Apparent trapped air
THKX	Equivalent thermal conductivity (x-direction)
THKY	Equivalent thermal conductivity (y-direction)
THKZ	Equivalent thermal conductivity (z-direction)
TMA	Total air mass
TMO	Total VOC mass
TMW	Total water mass
TO	Apparent trapped NAPL
UCn[(a)]	Solute flux (x-direction)
UG	Gas Darcy velocity (x-direction)
UGNC	Gas Darcy velocity (x-direction, node-centered)
UL	Aqueous Darcy velocity (x-direction)
ULNC	Aqueous Darcy velocity (x-direction, node-centered)
UN	NAPL Darcy velocity (x-direction)
UNNC	NAPL Darcy velocity (x-direction, node-centered)
US	Salt flux (x-direction)
USNC	Salt flux (x-direction, node-centered)
UQ	Energy flux (x-direction)
UQNC	Energy flux (x-direction, node-centered)

[(a)] n refers to a solute number

Table 7.1. Cont'd

Abbreviation	Definition
VCn[a]	Solute flux (y-direction)
VG	Gas Darcy velocity (y-direction)
VGNC	Gas Darcy velocity (y-direction, node-centered)
VL	Aqueous Darcy velocity (y-direction)
VLNC	Aqueous Darcy velocity (y-direction, node-centered)
VN	NAPL Darcy velocity (y-direction)
VNNC	NAPL Darcy velocity (y-direction, node-centered)
VS	Salt flux (y-direction)
VSNC	Salt flux (y-direction, node-centered)
VQ	Energy flux (y-direction)
VQNC	Energy flux (y-direction, node-centered)
WCn[a]	Solute flux (z-direction)
WG	Gas Darcy velocity (z-direction)
WGNC	Gas Darcy velocity (z-direction, node-centered)
WL	Aqueous Darcy velocity (z-direction)
WLNC	Aqueous Darcy velocity (z-direction, node-centered)
WN	NAPL Darcy velocity (z-direction)
WNNC	NAPL Darcy velocity (z-direction, node-centered)
WS	Salt flux (z-direction)
WSNC	Salt flux (z-direction, node-centered)
WQ	Energy flux (z-direction)
WQNC	Energy flux (z-direction, node-centered)
XGA	Air mass fraction in the gas
XGO	VOC mass fraction in the gas
XGW	Water mass fraction in the gas
XLA	Air mass fraction in the aqueous
XLO	VOC mass fraction in the aqueous
XLW	Water mass fraction in the aqueous
XMGA	Air mole fraction in the gas
XMGO	VOC mole fraction in the gas
XMGW	Water mole fraction in the gas
YGn[a]	Solute fraction in the gas
YLn[a]	Solute fraction in the aqueous
YNn[a]	Solute fraction in NAPL

[a] n refers to a solute number

Table 7.2. Surface Flux Column Title Abbreviations

Abbreviation	Definition
UGM	Gas mass flux rate (x-direction)
UGMS	Gas mass flux integral (x-direction)
ULM	Aqueous mass flux rate (x-direction)
ULMS	Aqueous mass flux integral (x-direction)
UNM	NAPL mass flux rate (x-direction)
UNMS	NAPL mass flux integral (x-direction)
UGV	Gas volumetric flux rate (x-direction)
UGVS	Gas volumetric flux integral (x-direction)
ULV	Aqueous volumetric flux rate (x-direction)
ULVS	Aqueous volumetric flux integral (x-direction)
UNV	NAPL volumetric flux rate (x-direction)
UNVS	NAPL volumetric flux integral (x-direction)
UQV	Energy flux rate (x-direction)
UQVS	Energy flux integral (x-direction)
US	Salt flux rate (x-direction)
USS	Salt flux integral (x-direction)
VGM	Gas mass flux rate (y-direction)
VGMS	Gas mass flux integral (y-direction)
VLM	Aqueous mass flux rate (y-direction)
VLMS	Aqueous mass flux integral (y-direction)
VNM	NAPL mass flux rate (y-direction)
VNMS	NAPL mass flux integral (y-direction)
VGV	Gas volumetric flux rate (y-direction)
VGVS	Gas volumetric flux integral (y-direction)
VLV	Aqueous volumetric flux rate (y-direction)
VLVS	Aqueous volumetric flux integral (y-direction)
VNV	NAPL volumetric flux rate (y-direction)
VNVS	NAPL volumetric flux integral (y-direction)
VQV	Energy flux rate (y-direction)
VQVS	Energy flux integral (y-direction)
VS	Salt flux rate (y-direction)
VSS	Salt flux integral (y-direction)
WGV	Gas volumetric flux rate (z-direction)
WGVS	Gas volumetric flux integral (z-direction)
WLV	Aqueous volumetric flux rate (z-direction)
WLVS	Aqueous volumetric flux integral (z-direction)
WNV	NAPL volumetric flux rate (z-direction)

Table 7.2. Cont'd

Abbreviation	Definition
WNVS	NAPL volumetric flux integral (z-direction)
WGM	Gas mass flux rate (z-direction)
WGMS	Gas mass flux integral (z-direction)
WLM	Aqueous mass flux rate (z-direction)
WLMS	Aqueous mass flux integral (z-direction)
WNM	NAPL mass flux rate (z-direction)
WNMS	NAPL mass flux integral (z-direction)
WQV	Energy flux rate (z-direction)
WQVS	Energy flux integral (z-direction)
WS	Salt flux rate (z-direction)
WSS	Salt flux integral (z-direction)

8.0 References

ANSI. 1978. *American National Standard Programming Language FORTRAN,* X3.9-1978, American National Standards Institute, Broadway, New York.

Nichols WE, NJ Aimo, M Oostrom, and MD White. 1997. *STOMP Subsurface Transport Over Multiple Phases Application Guide,* PNNL-11216, Pacific Northwest National Laboratory, Richland, Washington.

Reid RC, JM Prausnitz, and BE Poling. 1987. *The Properties of Gases and Liquids*, McGraw-Hill, New York.

Talbott S. 1988. *Managing Projects with make*, O'Reilly and Associates, Inc., Newton, Massachusetts.

White MD, and M Oostrom. 1996. *STOMP Subsurface Transport Over Multiple Phases Theory Guide,* PNNL-11217, Pacific Northwest National Laboratory, Richland, Washington.

Appendix

A. STOMP Input Control Card Formats

A.1 Simulation Title Card

Card Title[a] { ~Simulation [Title Card] }
Format: *Char*[a]

Version Number[a],
Format: *Integer*[a],

Simulation Title[a],
Format: *Char*[a],

User Name[a],
Format: *Char*[a],

Company Name[a],
Format: *Char*[a],

Input Creation Date[a],
Format: *Char*[a],

Input Creation Time[a],
Format: *Char*[a],

Number of Simulation Note Lines[a],
Format: *Integer*[a],

For: Number of Simulation Note Lines

Simulation Notes[a]
Format: *Char*[a] *(maximum of 132 characters per line)*

Endfor: Number of Simulation Note Lines

Endcard: Simulation Title Card

A.1.1 Simulation Title Card Examples

```
~Simulation Title Card
1,
Evaporation/Condensation Heat Pipe,
MD White,
Pacific Northwest Laboratory,
June 18 1994,
10:04 AM PDT,
4,
This application problem follows the heat-pipe problem solved
semi-analytically by Udell and Fitch.  The soil moisture retention
function has been changed to a modified van Genuchten function to
allow saturations for all matric suctions.
```

```
~Simulation Title Card
1,
Field Test #12,
MD White,
PNNL,
Monday April 5 1996,
12:34,
2,
Simulation of field test #12 at Edwards AFB.
Starting time 1/3/96 14:19, Ending time 1/8/96 9:19.
```

```
~Simulation Title Card
1,
Henry's Problem for Salt Water Intrusion,
MD White,
Pacific Northwest Laboratory,
August 9 1995,
8:30:00 AM PDT,
1,
STOMP Application Guide Problem 4.1
```

A.2 Solution Control Card

Card Title[a] {~Solution [Control Card] }
Format: *Char*[a]

Execution Mode Option[a],
If: Operational Mode Option = { Water }
{ Normal [{ No Flow } { Dynamic Domain }] }
{ Restart [{ No Flow } { Dynamic Domain }] [Zero Solutes] }
{ Initial Conditions }
Else:
{ Normal }
{ Restart }
{ Initial Conditions }
Endif:
Format: *Char*[a],

Operational Mode Option[a],
{ Water [[TVD] Transport] }
{ Water-Air [[TVD] Transport] }
{ Water-Air-Energy [[TVD] Transport] [Ice] }
{ Water-Oil [[TVD] Transport] }
{ Water-Oil-Dissolved Oil [LFL] [[TVD] Transport] }
{ Water-Oil-Dissolved Oil-Surfactant [LFL] [[TVD] Transport] }
{ Water-Oil-Air [[TVD] Transport] }
{ Water-Oil-Air-Energy [[TVD] Transport] }
{ Water-Salt [LFL] [[TVD] Transport] }
{ Water-Air-Salt [LFL] [[TVD] Transport] }
{ Water-Air-Energy-Salt [LFL] [[TVD] Transport] [Ice] }
Format: *Char*[a],

If: Execution Mode Option = { Initial Conditions }
Endcard: Solution Control Card
Endif:

Number of Execution Time Periods[a],
Format: *Integer*[a],

Solution Control Card (contd)

For: Number of Execution Time Periods

If: Execution Mode Option = { Normal }
Initial Time[a], Units[b] (s), Final Time[c], Units[d] (s),
Initial Time Step[e], Units[f] (s),
Maximum Time Step[g], Units[h] (s),
Time Step Acceleration Factor[i],
Maximum Number of Newton-Raphson Iterations[j],
Convergence Criterion[k],
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], Real[i], Integer[j], Real[k],*
ElseIf: Execution Mode Option = { Restart }
Initial Time[a], Units[b] (s), Final Time[c], Units[d] (s),
Initial Time Step[e], Units[f] (s),
Maximum Time Step[g], Units[h] (s),
Time Step Acceleration Factor[i],
Maximum Number of Newton-Raphson Iterations[j],
Convergence Criterion[k],
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], Real[i], Integer[j], Real[k],*
Endif:

Endfor: Number of Execution Time Periods

Maximum CPU Time[a], Units[b] (s),
Maximum Clock Time[c], Units[d] (s),
Maximum Number of Time Steps[e]
Format: *Real[a], Char[b], Real[c], Char[d],Integer[e],*

If: Operational Mode Option = { Water-Air } { Water-Air-Energy }
Aqueous Diffusion Option[a],
{ Zero }
{ Constant }
{ Variable }
If: Aqueous Diffusion Option = { Constant }
Dissolved Air Diffusion Coefficient[b], Unit[c] (m^2/s),
Format: *Char[a], Real[b], Char[c],*
Else:
Format: *Char[a],*
Endif:
Endif:

Solution Control Card (contd)

If: Operational Mode Option = { Water-Oil } { Water-Oil-Dissolved Oil }
{ Water-Oil-Dissolved Oil-Surfactant }
Aqueous Diffusion Option[a],
{ Zero }
{ Constant }
{ Variable }
If: Aqueous Diffusion Option = { Constant }
Dissolved Oil Diffusion Coefficient[b], Unit[c] (m^2/s),
Format: *Char[a], Real[b], Char[c],*
Else:
Format: *Char[a],*
Endif:
Endif:

If: Operational Mode Option = { Water-Oil-Air } { Water-Oil-Air-Energy }
Aqueous Diffusion Option[a],
{ Zero }
{ Constant }
{ Variable }
If: Aqueous Diffusion Option = { Constant }
Dissolved Air Diffusion Coefficient[b], Unit[c] (m^2/s),
Dissolved Oil Diffusion Coefficient[d], Unit[e] (m^2/s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e],*
Else:
Format: *Char[a],*
Endif:
Endif:

If: Operational Mode Option = { Water-Air } { Water-Air-Energy }
Gas Diffusion Option[b],
{ Zero }
{ Constant }
{ Variable }
{ Enhanced }
If: Gas Diffusion Option = { Constant }
Water Vapor Diffusion Coefficient[b], Unit[c] (m^2/s),
Format: *Char[a], Real[b], Char[c],*
Elseif: Gas Diffusion Option = { Enhanced }
Clay Mass Fraction[b],
Format: *Char[a], Real[b],*
Endif:
Endif:

Solution Control Card (contd)

If: Operational Mode Option = { Water-Oil-Air } { Water-Oil-Air-Energy }
Gas Diffusion Option[b],
{ Zero }
{ Constant }
{ Variable }
{ Enhanced }
If: Gas Diffusion Option = { Constant }
Water Vapor Diffusion Coefficient[b], Unit[c] (m^2/s),
Oil Vapor Diffusion Coefficient[d], Units[e] (m^2/s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e],*
Elseif: Gas Diffusion Option = { Enhanced }
Clay Mass Fraction[b],
Format: *Char[a], Real[b],*
Endif:
Endif:

Number of Interfacial Averaging Variables[a]
Format: *Integer[a],*

For: Number of Interfacial Averaging Variables

Surface Variable Option[a],
{ Aqueous Density } { Aqueous Relative Permeability }
{ Aqueous Viscosity } { Dissolved Air Diffusion }
{ Dissolved Salt Diffusion } { Dissolved Oil Diffusion }
{ Gas Density } { Gas Relative Permeability }
{ Gas Viscosity } { Hydraulic Dispersion }
{ Intrinsic Permeability } { NAPL Density }
{ NAPL Relative Permeability } { NAPL Viscosity }
{ Solute Diffusion } { Thermal Conductivity }
{ Oil Vapor Diffusion } { Water Vapor Diffusion }
Interfacial Averaging Scheme Option[b]
{ Harmonic } { Geometric } { Arithmetic } { Upwind }
Format: *Char[a], Char[b],*

Endfor: Number of Interfacial Averaging Variables

Endcard: Solution Control Card

A.2.1 Solution Control Card Examples

```
~Solution Control Card
Normal,
Water-Air-Energy,
1,
0,day,876.6,day,10,S,100,day,1.25,16,1.E-06,
1,day,1,day,1000,
Variable Aqueous Diffusion,
Variable Gas Diffusion,
# Interfacial average defaults
3,
Gas Relative Permeability, Arithmetic,
Aqueous Relative Permeability, Harmonic,
Intrinsic Permeability, Geometric,
```

```
~Solution Control Card
Normal,
Water-Oil-Dissolved Oil-Surfactant,
# Two execution periods specified
2,
0,s,0.25,hr,0.1,s,0.005,hr,1.25,16,1.e-06,
0.25,s,2.3769,hr,0.005,hr,0.01,hr,1.2,16,1.e-06,
1,day,1,day,10000,
Variable,
0,
```

```
~Solution Control Card
Normal,
Water w/ Solute Transport,
1,
0,min,6816.95,min,1,s,20,min,1.25,8,1.e-06,
1,hr,1,hr,1000,
0,
```

```
~Solution Control Card
Normal,
Water-Salt,
1,
0.0,yr,1.e+3,yr,1,s,1.e+3,yr,1.25,8,1.e-6,
1,hr,1,hr,1000,
0,
```

A.3 Grid Card

Card Title[a] { ~Grid [Card] }
Format: *Char[a]*

Coordinate System Option[a],
{ Cartesian }
{ Cylindrical }
{ Uniform Cartesian }
{ Uniform Cylindrical }
{ Tilted Cartesian }

If: Coordinate System Option ≠ { Tilted Cartesian }
Format: *Char[a]*,
Elseif:
X-Z Plane Tilt Angle[b], Units[c] (deg),
Y-Z Plane Tilt Angle[d], Units[e] (deg),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e],*
Endif:

Number of X-Dir. Nodes[a],
Number of Y-Dir. Nodes[b],
Number of Z-Dir. Nodes[c],
Format: *Integer[a], Integer[b], Integer[c],*

If: Coordinate System Option = { Cartesian }

For: Number of X-Dir. Nodes
< Surface Position[a], Units[b] (m), > or < Count[a] @ Node Width[b], Units[c] (m), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of X-Dir. Nodes

For: Number of Y-Dir. Nodes
< Surface Position[a], Units[b] (m), } or < Count[a] @ Node Width[b], Units[c] (m), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of Y-Dir. Nodes

For: Number of Z-Dir. Nodes
< Surface Position[a], Units[b] (m), > or < Count[a] @ Node Width[b], Units[c] (m), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of Z-Dir. Nodes

Grid Card (contd)

Elseif: Coordinate System Option = { Cylindrical }

For: Number of Radial-Dir. Nodes
< Surface Position[a], Units[b] (m), > or < Count[a] @ Node Width[b], Units[c] (m), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of Radial-Dir. Nodes

For: Number of Azimuthal-Dir. Nodes
< Surface Position[a], Units[b] (deg), > or < Count[a] @ Node Width[b], Units[c] (deg), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of Azimuthal-Dir. Nodes

For: Number of Z-Dir. Nodes
< Surface Position[a], Units[b] (m), > or < Count[a] @ Node Width[b], Units[c] (m), >
Format: < *Real[a], Char[b],* > or < *Integer[a]@Real[a], Char[b],* >
Endfor: Number of Z-Dir. Nodes

Elseif: Coordinate System Option = { Uniform Cartesian }

X-Dir. Node Dimension[a], Units[b] (m)
Format: *Real[a], Char[b],*

Y-Dir. Node Dimension[a], Units[b] (m)
Format: *Real[a], Char[b],*

Z-Dir. Node Dimension[a], Units[b] (m)
Format: *Real[a], Char[b],*

Elseif: Coordinate System Option = { Uniform Cylindrical }

Radial-Dir. Node Dimension[a], Units[b] (m)
Format: *Real[a], Char[b],*

Azimuthal-Dir. Node Dimension[a], Units[b] (deg)
Format: *Real[a], Char[b],*

Z-Dir. Node Dimension[a], Units[b] (m)
Format: *Real[a], Char[b],*

Endif:

Endcard: Grid Card

A.3.1 Grid Card Examples

```
#----------------------
~Grid Card
#----------------------
Uniform Cartesian,
20,1,10,
10,cm,
10,cm,
10,cm,
```

```
~Grid Card
Cylindrical,
50,1,113,
#Nonuniform grid spacing
0,in,3.125,in,5.125,in,8,in,12,in,18@6,in,10@12,in,10@24,in,8@48,in,
0,deg,45,deg,
0,in,113@6,in,
```

```
~Grid Card
Uniform Cartesian,
1,1,92,
4.25388924,cm,
4.25388924,cm,
0.125,cm,
```

A.4 Inactive Nodes Card

Card Title[a] { ~Inactive [Nodes Card] }
Format: *Char*[a]

Number of Inactive Node Domains[a],
Format: *Integer*[a],

For: Number of Inactive Node Domains

I-Start Index[a], I-End Index[b],
J-Start Index[c], J-End Index[d],
K-Start Index[e], K-End Index[f],
Format: *Integer*[a], *Integer*[b], *Integer*[c], *Integer*[d], *Integer*[e], *Integer*[f],

Endfor: Number of Inactive Node Domains

Endcard: Inactive Nodes Card

A.4.1 Inactive Nodes Card Examples

```
~Inactive Nodes Card
4,
1,1,1,1,14,113,
2,2,1,1,14,33,
2,2,1,1,101,113,
2,2,1,1,70,70,
```

```
#----------
~Inactive
#----------
1,
2,2,1,1,14,20
```

```
#
~Inactive Nodes Card
#
0,
# No inactive nodes
```

A.5 Rock/Soil Zonation Card

Card Title[a] { ~Rock/Soil [Zonation Card] }
Format: *Char[a]*

Number of Rock/Soil Zonation Domains[a],
Format: *Integer[a]*,

For: Number of Rock/Soil Zonation Domains
Rock/Soil Name[a],
I-Start Index[b], I-End Index[c],
J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g],*
Endfor: Number of Rock/Soil Zonation Domains

Endcard: Rock/Soil Zonation Card

A.5.1 Rock/Soil Zonation Card Examples

```
~Rock/Soil Zonation Card
1,
20/30 Ottawa Sand,1,1,1,1,1,92,
```

```
~Rock/Soil Zonation Card
7,
SP1,1,50,1,1,86,113,
SP2,1,50,1,1,63,85,
SM-ML1,1,50,1,1,69,69,
SM-SP1,1,50,1,1,46,62,
SP3,1,50,1,1,16,45,
SW1,1,50,1,1,1,15,
US,2,2,1,1,71,100,
```

A.6 Mechanical Properties Card

Card Title[a] { ~Mechanical [Properties Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

If: Operational Mode Option: Solution Control Card = { Water w/o Transport }
Rock/Soil Name[a],
Particle Density[b] (2650.0), Units[c] (kg/m^3),
If: Rock/Soil Name = {{ Fractured }}
Matrix Total Porosity[d], Matrix Diffusive Porosity[e],
Fracture Total Porosity[f], Fracture Diffusive Porosity[g],
Specific Storativity[h], Units[i] (1/m),
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Real*[g], *Real*[h], *Char*[i],
Else:
Total Porosity[d], Diffusive Porosity[e],
Specific Storativity[f], Units[g] (1/m),
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Char*[g],
Endif:

Elseif: Operational Mode Option: Solution Control Card = { Water w/ Transport }
{ Water-Salt }
Rock/Soil Name[a],
Particle Density[b] (2650.0), Units[c] (kg/m^3),
If: Rock/Soil Name = {{ Fractured }}
Matrix Total Porosity[d], Matrix Diffusive Porosity[e],
Fracture Total Porosity[f], Fracture Diffusive Porosity[g],
Specific Storativity[h], Units[i] (1/m),
Tortuosity Function Option[j],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[k],
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Real*[g], *Real*[h], *Char*[i], *Char*[j], *Real*[k],
Else:
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Real*[g], *Real*[h], *Char*[i], *Char*[j],
Endif:
Else:
Total Porosity[d], Diffusive Porosity[e],
Specific Storativity[f], Units[g] (1/m),
Tortuosity Function Option[h],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[i],
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Char*[g], *Char*[h], *Real*[i],
Else:
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Real*[e], *Real*[f], *Char*[g], *Char*[h],
Endif:
Endif:

Mechanical Properties Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air } { Water-Air-Energy } { Water-Air-Salt } { Water-Air-Energy-Salt }
Rock/Soil Name[a],
Particle Density[b] (2650.0), Units[c] (kg/m^3),
If: Rock/Soil Name = {{ Fractured }}
Matrix Total Porosity[d], Matrix Diffusive Porosity[e],
Fracture Total Porosity[f], Fracture Diffusive Porosity[g],
Specific Storativity[h], Units[i] (1/m),
Tortuosity Function Option[j],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[k], Gas-Phase Tortuosity[l],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Real[g], Real[h], Char[i], Char[j], Real[k], Real[l],*
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Real[g], Real[h], Char[i], Char[j],*
Endif:
Else:
Total Porosity[d], Diffusive Porosity[e],
Specific Storativity[f], Units[g] (1/m),
Tortuosity Function Option[h],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[i], Gas-Phase Tortuosity[j],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Char[g], Char[h], Real[i], Real[j],*
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Char[g], Char[h],*
Endif:
Endif:

Mechanical Properties Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil } { Water-Oil-Dissolved Oil } { Water-Oil-Dissolved Oil-Surfactant }
Rock/Soil Name[a],
Particle Density[b] (2650.0), Units[c] (kg/m^3),
If: Rock/Soil Name = {{ Fractured }}
Matrix Total Porosity[d], Matrix Diffusive Porosity[e],
Fracture Total Porosity[f], Fracture Diffusive Porosity[g],
Specific Storativity[h], Units[i] (1/m),
Tortuosity Function Option[j],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[k], NAPL-Phase Tortuosity[l],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Real[g], Real[h], Char[i], Char[j], Real[k], Real[l],*
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Real[g], Real[h], Char[i], Char[j],*
Endif:
Else:
Total Porosity[d], Diffusive Porosity[e],
Specific Storativity[f], Units[g] (1/m),
Tortuosity Function Option[h],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[i], NAPL-Phase Tortuosity[j],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Char[g], Char[h], Real[i], Real[j],*
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Char[g], Char[h],*
Endif:
Endif:

Mechanical Properties Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
{ Water-Oil-Air-Energy }
Rock/Soil Name[a],
Particle Density[b] (2650.0), Units[c] (kg/m^3),
If: Rock/Soil Name = {{ Fractured }}
Matrix Total Porosity[d], Matrix Diffusive Porosity[e],
Fracture Total Porosity[f], Fracture Diffusive Porosity[g],
Specific Storativity[h], Units[i] (1/m),
Tortuosity Function Option[j],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[k], Gas-Phase Tortuosity[l], NAPL-Phase Tortuosity[m],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f],*
Real[g], Real[h], Char[i], Char[j], Real[k], Real[l], Real[m],
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Real[g], Real[h], Char[i], Char[j],*
Endif:
Else:
Total Porosity[d], Diffusive Porosity[e],
Specific Storativity[f], Units[g] (1/m),
Tortuosity Function Option[h],
{ Constant } { Millington and Quirk }
If: Tortuosity Function Option = { Constant }
Aqueous-Phase Tortuosity[i], Gas-Phase Tortuosity[j], NAPL-Phase Tortuosity[k],
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f],*
Char[g], Char[h], Real[i], Real[j], Real[k],
Else:
Format: *Char[a], Real[b], Char[c], Real[d], Real[e], Real[f], Char[g], Char[h],*
Endif:
Endif:
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Mechanical Properties Card

A.6.1 Mechanical Properties Card Examples

```
~Mechanical Properties Card
SP1,2.63,g/cm^3,0.19,0.19,,,,
SP2,2.63,g/cm^3,0.24,0.24,,,,
SM-ML1,2.63,g/cm^3,0.35,0.35,,,,
SM-SP1,2.63,g/cm^3,0.37,0.37,,,,
SP3,2.63,g/cm^3,0.27,0.27,,,,
SW1,2.63,g/cm^3,0.28,0.28,,,,
US,2.63,g/cm^3,0.96,0.96,,,,
```

```
~Mechanical Properties Card
Sand1,2650,kg/m^3,0.34,0.34,0,,Millington and Quirk,
```

```
~Mechanical Properties Card
Sand,2650,kg/m^3,0.4,0.4,,,Constant,0.5,0.5,
```

```
~Mechanical Properties Card
Geologic Media,,,0.35,0.35,,,Constant Diffusion,1.0,
```

```
~Mechanical Properties Card
20/30 Ottawa Sand,2650,kg/m^3,0.3431,0.3431,0,1/m,Millington and Quirk,
```

A.7 Hydraulic Properties Card

Card Title[a] { ~Hydraulic [Properties Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

Rock/Soil Name[a],
If: Rock/Soil Name = {{ Fractured }}
X-Dir. (Radial-Dir.) Matrix Intrinsic Permeability[b], Units[c] (m^2),
or X-Dir. (Radial-Dir.) Matrix Hydraulic Conductivity[b], Units[c] (hc m/s),

Y-Dir. (Azimuthal-Dir.) Matrix Intrinsic Permeability[d], Units[e] (m^2),
or Y-Dir. (Azimuthal-Dir.) Matrix Hydraulic Conductivity[d], Units[e] (hc m/s),

Z-Dir. Matrix Intrinsic Permeability[f], Units[g] (m^2),
or Z-Dir. Matrix Hydraulic Conductivity[f], Units[g] (hc m/s),

X-Dir. (Radial-Dir.) Fracture Intrinsic Permeability[h], Units[i] (m^2),
or X-Dir. (Radial-Dir.) Fracture Hydraulic Conductivity[h], Units[i] (hc m/s),

Y-Dir. (Azimuthal-Dir.) Fracture Intrinsic Permeability[j], Units[l] (m^2),
or Y-Dir. (Azimuthal-Dir.) Fracture Hydraulic Conductivity[j], Units[k] (hc m/s),

Z-Dir. Fracture Intrinsic Permeability[l], Units[m] (m^2),
or Z-Dir. Fracture Hydraulic Conductivity[l], Units[m] (hc m/s),

Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Char*[g],
Real[h], *Char*[i], *Real*[j], *Char*[k], *Real*[l], *Char*[m],
Else:
X-Dir. (Radial-Dir.) Intrinsic Permeability[b], Units[c] (m^2),
or X-Dir. (Radial-Dir.) Hydraulic Conductivity[b], Units[c] (hc m/s),

Y-Dir. (Azimuthal-Dir.) Intrinsic Permeability[d], Units[e] (m^2),
or Y-Dir. (Azimuthal-Dir.) Hydraulic Conductivity[d], Units[e] (hc m/s),

Z-Dir. Intrinsic Permeability[f], Units[g] (m^2),
or Z-Dir. Hydraulic Conductivity[f], Units[g] (hc m/s),

Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Char*[g],
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Hydraulic Properties Card

A.7.1 Hydraulic Properties Card Examples

```
~Hydraulic Properties Card
20/30 Ottawa Sand,1.6e-7,cm^2,1.6e-7,cm^2,1.6e-7,cm^2,
```

```
~Hydraulic Properties Card
Geologic Media,1.020408e-9,m^2,,,1.020408e-9,m^2,
```

```
~Hydraulic Properties Card
Sand1,,,,,7.5,hc m/day,
```

```
~Hydraulic Properties Card
SP1,4.1987e-12,m^2,,,4.1987e-12,m^2,
SP2,9.3436e-13,m^2,,,9.3436e-13,m^2,
SM-ML1,5.3223e-13,m^2,,,5.3223e-13,m^2,
SM-SP1,7.695e-12,m^2,,,5.13e-12,m^2,
SP3,5.505e-12,m^2,,,3.67e-12,m^2,
SW1,9.195e-12,m^2,,,6.13e-12,m^2,
US,1.0e+04,hc cm/hr,,,1.0e+06,hc cm/hr,
```

```
~Hydraulic Properties Card
Sand,5.0e-4,hc ft/s,5.0e-4,hc ft/s,5.0e-4,hc ft/s,
```

A.8 Thermal Properties Card

Card Title[a] { ~Thermal [Properties Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

Rock/Soil Name[a],
Thermal Conductivity Function Option[b],
{ Constant } { Parallel } { Linear } { Somerton } { Campbell }

If: Thermal Conductivity Function Option = { Constant }
X-Dir. Thermal Conductivity[c], Units[d] (W/m K),
Y-Dir. Thermal Conductivity[e], Units[f] (W/m K),
Z-Dir. Thermal Conductivity[g], Units[h] (W/m K),
Specific Heat[i], Units[j] (J/kg K),
Format: *Char*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Char*[h], *Real*[i], *Char*[j],
Elseif: Thermal Conductivity Function Option = { Parallel }
X-Dir. Rock/Soil Grain Thermal Conductivity[c], Units[d] (W/m K),
Y-Dir. Rock/Soil Grain Thermal Conductivity[e], Units[f] (W/m K),
Z-Dir. Rock/Soil Grain Thermal Conductivity[g], Units[h] (W/m K),
Specific Heat[h], Units[i] (J/kg K),
Format: *Char*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Char*[h], *Real*[i], *Char*[j],
Elseif: Thermal Conductivity Function Option = { Linear }
X-Dir. Rock/Soil Unsaturated Thermal Conductivity[c], Units[d] (W/m K),
Y-Dir. Rock/Soil Unsaturated Thermal Conductivity[e], Units[f] (W/m K),
Z-Dir. Rock/Soil Unsaturated Thermal Conductivity[g], Units[h] (W/m K),
X-Dir. Rock/Soil Water Saturated Thermal Conductivity[i], Units[j] (W/m K),
Y-Dir. Rock/Soil Water Saturated Thermal Conductivity[k], Units[l] (W/m K),
Z-Dir. Rock/Soil Water Saturated Thermal Conductivity[m], Units[n] (W/m K),
Specific Heat[o], Units[p] (J/kg K),
Format: *Char*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Char*[h],
Real[i], *Char*[j], *Real*[k], *Char*[l], *Real*[m], *Char*[n], *Real*[o], *Char*[p],
Elseif: Thermal Conductivity Function Option = { Somerton }
X-Dir. Rock/Soil Unsaturated Thermal Conductivity[c], Units[d] (W/m K),
Y-Dir. Rock/Soil Unsaturated Thermal Conductivity[e], Units[f] (W/m K),
Z-Dir. Rock/Soil Unsaturated Thermal Conductivity[g], Units[h] (W/m K),
X-Dir. Rock/Soil Water Saturated Thermal Conductivity[i], Units[j] (W/m K),
Y-Dir. Rock/Soil Water Saturated Thermal Conductivity[k], Units[l] (W/m K),
Z-Dir. Rock/Soil Water Saturated Thermal Conductivity[m], Units[n] (W/m K),
Specific Heat[o], Units[p] (J/kg K),
Format: *Char*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Char*[h],
Real[i], *Char*[j], *Real*[k], *Char*[l], *Real*[m], *Char*[n], *Real*[o], *Char*[p],
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Thermal Properties Card

A.8.1 Thermal Properties Card Examples

```
~Thermal Properties Card
Sand,Somerton,0.582,W/m K,,,,,1.13,W/m K,,,,,700,J/kg K,
```

A.9 Saturation Function Card

Card Title[a] { ~Saturation Function [Card] }
Format: *Char*[a]

If: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
{ Water-Oil-Dissolved Oil-Surfactant }
Fluid Pair Interfacial Tension Option
{ Gas-Aqueous Interfacial Tension[a], Units[b] (N/m),
NAPL-Aqueous Interfacial Tension[c], Units[d] (N/m),
Null[e], Null[f], }

{ Null[a], Null[b],
NAPL-Aqueous Interfacial Tension[c], Units[d] (N/m),
Gas-NAPL Interfacial Tension[e], Units[f] (N/m), }

{ Gas-Aqueous Interfacial Tension[a], Units[b] (N/m),
Null[c], Null[d],
Gas-NAPL Interfacial Tension[e], Units[f] (N/m), }
Format: *Real*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f],
Endif:

Saturation Function Card (contd)

For: Number of Rock/Soil Types

If: Operational Mode Option: Solution Control Card = { Water } { Water-Air } or { Water-Air-Energy w/o Ice }
Rock/Soil Name[a],
Saturation Function[b],
{ [Nonhysteretic] van Genuchten } { [Nonhysteretic] Brooks and Corey }
{ Hysteretic van Genuchten } { Hysteretic Brooks and Corey }
{ Haverkamp } { Tabular [Water Content] }

If: Saturation Function Option = { [Nonhysteretic] van Genuchten }

If: Rock/Soil Name = {{ Fractured }}
Matrix "α" Parameter[c], Units[d] (1/m)', Matrix "n" Parameter[e],
Matrix Minimum Saturation[f],
Fracture "α" Parameter[g], Units[h] (1/m)', Fracture "n" Parameter[i],
Fracture Minimum Saturation[j],
Matrix "m" Parameter[k], Fracture "m" Parameter[l],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f],*
Real[g], Char[h], Real[i], Real[j], Real[k], Real[l],

Else:
"α" Parameter[c], Units[d] (1/m)', "n" Parameter[e],
Minimum Saturation[f], "m" Parameter[g],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g],*

Endif:

Elseif: Saturation Function Option = { [Nonhysteretic] Brooks and Corey }

If: Rock/Soil Name = {{ Fractured }}
Matrix Entry Head[c], Units[d] (m), Matrix "λ" Parameter[e],
Matrix Minimum Saturation[f],
Fracture Entry Head[g], Units[h] (m), Fracture "λ" Parameter[i],
Fracture Minimum Saturation[j],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h], Real[i], Real[j],*

Else:
Entry Head[c], Units[d] (m), "λ" Parameter[e], Minimum Saturation[f],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f],*

Endif:

Elseif: Saturation Function Option = { Hysteretic van Genuchten }
"α" Drainage[c], Units[d] (1/m), "α" Imbibition[e], Units[f] (1/m),
"n" Parameter[g], Minimum Saturation[h],
Maximum Effective Air Residual Saturation - Air/Water[i],
Initial Hysteretic Branch Index Option[j],
{ Main Drainage } { Main Imbibition } { Drainage Scanning Curve }
Number of Scanning Paths[k],
" m" Parameter[l]
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Char[f],*
Real[g], Real[h], Real[i], Char[j], Integer[k], Real[l],

Elseif: Saturation Function Option = { Hysteretic Brooks and Corey }
Entry Head Drainage[c], Units[d] (m), Entry Head Imbibition[e], Units[f] (m),
"λ" Parameter[g], Minimum Saturation[h],
Maximum Effective Air Residual Saturation - Air/Water[i],
Initial Hysteretic Branch Index Option[j]
{ Main Drainage } { Main Imbibition } { Drainage Scanning Curve }
Number of Scanning Paths[k]
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Real[h], Real[i], Char[j], Integer[k],*

Saturation Function Card (contd)

Elseif: Saturation Function Option = { Haverkamp }
"ψ" Parameter[c], Units[d] (m), "α" Parameter[e], "β" Parameter[f], Minimum Saturation[g],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g],*
Elseif: Saturation Function Option = { Tabular }
Number of Table Entries[a],
Format: Char[a], Char[b], Integer[a]
For: Number of Table Entries
Air-Water Capillary Head[a], Units[b] (m), Aqueous Saturation[c],
Format: *Real[a], Char[b], Real[c],*
Endfor: Number of Table Entries
Endif:
Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy w/ Ice }
Rock/Soil Name[a],
Saturation Function[b],
{ [Nonhysteretic] van Genuchten }
{ [Nonhysteretic] Brooks and Corey }
If: Saturation Function Option = { [Nonhysteretic] van Genuchten }
"α" Parameter[c], Units[d] (1/m)',
"n" Parameter[e],
Minimum Saturation[f],
Air/Water Scaling Factor[g],
Ice/Water Scaling Factor[h],
" m" Parameter[i],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Real[h], Real[i],*
Elseif: Saturation Function Option = { [Nonhysteretic] Brooks and Corey }
Entry Head[c], Units[d] (m),
"λ" Parameter[e],
Minimum Saturation[f],
Air/Water Scaling Factor[g],
Ice/Water Scaling Factor[h],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Real[h],*
Endif:

Saturation Function Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil } { Water-Oil-Air } {Water-Oil-Air-Energy }
Rock/Soil Name[a],
Saturation Function[b]
{ [Nonhysteretic] van Genuchten }
{ [Nonhysteretic] Brooks and Corey }
{ Hysteretic van Genuchten }
{ Hysteretic Brooks and Corey }
If: Saturation Function Option = { [Nonhysteretic] van Genuchten }
"α" Parameter[c], Units[d] (1/m),
"n" Parameter[e],
Minimum Saturation[f],
Air/Water Scaling Factor[g],
Oil/Water Scaling Factor[h],
Air/Oil Scaling Factor[i],
" m" Parameter[j],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Real[h], Real[i], Real[j],*
Elseif: Saturation Function Option = { [Nonhysteretic] Brooks and Corey }
Entry Head[c], Units[d] (m),
"λ" Parameter[e],
Minimum Saturation[f],
Air/Water Scaling Factor[g],
Oil/Water Scaling Factor[h],
Air/Oil Scaling Factor[i],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Real[h], Real[i],*
Elseif: Saturation Function Option = { Hysteretic van Genuchten }
"α" Drainage Parameter[c], Units[d] (1/m), "α" Imbibition Parameter[e], Units[f] (1/m),
"n" Parameter[g],
Minimum Saturation[h],
Air/Water Scaling Factor[i],
Oil/Water Scaling Factor[j],
Air/Oil Scaling Factor[k],
Maximum Effective Air Residual Saturation - Air/Water[l],
Maximum Effective Air Residual Saturation - Air/Oil[m],
Maximum Effective Oil Residual Saturation - Oil/Water[n],
Initial Hysteretic Branch Index° { Main Drainage, Main Imbibition, Drainage Scanning Curve },
Number of Scanning Paths[p],
" m" Parameter[q]
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Real[h], Real[i], Real[j], Real[k], Real[l], Real[m], Real[n], Char[o], Integer[p], Real[q],*
Endif:

Saturation Function Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
{ Water-Oil-Dissolved Oil-Surfactant }
Rock/Soil Name[a],
Saturation Function[b]
{ [Nonhysteretic] van Genuchten }
{ [Nonhysteretic] Brooks and Corey }
{ [NAPL] Entrapment van Genuchten }
{ [NAPL] Entrapment Brooks and Corey }
If: Saturation Function Option = { [Nonhysteretic] van Genuchten }
"α" Parameter[c], Units[d] (1/m),
"n" Parameter[e],
Minimum Saturation[f],
Reference Fluid Pair Interfacial Tension[g], Units[h] (N/m),
" m" Parameter[i],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h], Real[i],*
Elseif: Saturation Function Option = { [Nonhysteretic] Brooks and Corey }
Entry Head[c], Units[d] (m),
"λ" Parameter[e],
Minimum Saturation[f],
Reference Fluid Pair Interfacial Tension[g], Units[h] (N/m),
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h],*
Elseif: Saturation Function Option = { [NAPL] Entrapment van Genuchten }
"α" Parameter[c], Units[d] (1/m),
"n" Parameter[e],
Minimum Saturation[f],
Reference Fluid Pair Interfacial Tension[g], Units[h] (N/m),
NAPL Effective Residual Saturation[i],
Critical Trapping Number[j],
" m" Parameter[k],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h], Real[i], Real[j], Real[k],*

Elseif: Saturation Function Option = { [NAPL] Entrapment Brooks and Corey }
Entry Head[c], Units[d] (m),
"λ" Parameter[e],
Minimum Saturation[f],
Reference Fluid Pair Interfacial Tension[g], Units[h] (N/m),
NAPL Effective Residual Saturation[i],
Critical Trapping Number[j],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h], Real[i], Real[j],*
Endif:
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Saturation Function Card

A.9.1 Saturation Function Card Examples

```
~Saturation Function Card
Sand1,Nonhysteretic Van Genuchten,0.0283,1/cm,7.6,0.32,1.0,1.8,2.25,,
```

```
~Saturation Function Card
Sand,Nonhysteretic Van Genuchten,1.563,1/m,5.4,0.15,,
```

```
~Saturation Function Card
Geologic Media,Nonhysteretic van Genuchten,0.2,1/cm,1.8,0.0,,
```

```
~Saturation Function Card
72.0,dynes/cm,47.8,dynes/cm,,,
20/30 Ottawa Sand,Entrapment van Genuchten,1.885,1/cm,5.359, ¬
   0.1,72.0,dynes/cm,0.12665,5.e-5,,1.0,
```

```
~Saturation Function Card
SP1,Nonhysteretic Brooks and Corey,4.32,cm,0.528,,
SP2,Nonhysteretic Brooks and Corey,6.41,cm,0.415,,
SM-ML1,Brooks and Corey,10.0,cm,0.383,,
SM-SP1,Brooks and Corey,9.0,cm,0.430,,
SP3,Nonhysteretic Brooks and Corey,10.7,cm,0.494,,
SW1,Nonhysteretic Brooks and Corey,1.17,cm,0.346,,
```

A.10 Aqueous Relative Permeability Card

Card Title[a] { ~Aqueous Rel [ative Permeability Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

If: Operational Mode Option: Solution Control Card = { Water } { Water-Air } { Water-Air-Energy } { Water-Salt } { Water-Air-Salt } { Water-Air-Energy-Salt }
Rock/Soil Name[a],
Permeability Function Option[b],
{ Constant } { Mualem } { Burdine } { Fatt and Klikoff } { Corey } { Haverkamp } {Tauma and Vauclin } { Tabular { w/ Water Content } }
If: Permeability Function Option = { Constant }
If: Rock/Soil Name = {{ Fractured }}
Matrix Aqueous Relative Permeability[c],
Fracture Aqueous Relative Permeability[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],
Else:
Aqueous Relative Permeability[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Endif:
Elseif: Permeability Function Option = { Mualem } { Burdine }
If: Saturation Function Option = {{ van Genuchten }}
van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ Brooks and Corey }}
Brooks and Corey "λ" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ van Genuchten }}
and Rock/Soil Name = {{ Fractured }}
Matrix van Genuchten "m" parameter[c],
Fracture van Genuchten "m" parameter[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],
Elseif: Saturation Function Option = {{ Brooks and Corey }}
and Rock/Soil Name = {{ Fractured }}
Matrix Brooks and Corey "λ" parameter[c],
Fracture Brooks and Corey "λ" parameter[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],
Endif:
Elseif: Permeability Function Option = { Fatt and Klikoff } { Corey }
Format: *Char*[a], *Char*[b],
Elseif: Permeability Function Option = { Haverkamp }
"a" Parameter[c], "γ" Parameter[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],
Elseif: Permeability Function Option = { Tauma and Vauclin }
"α" Parameter[c], "β" Parameter[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],

Aqueous Relative Permeability Card (contd)

Elseif: Permeability Function Option = { Tabular } and { Content }
For: Number of Table Entries
Aqueous Moisture Content[a], Aqueous Relative Permeability[b],
Format: *Real*[a], *Real*[b],
Endfor: Number of Table Entries
Elseif: Permeability Function Option = { Tabular }
Number of Table Entries[c],
Format: *Char*[a], *Char*[b], *Integer*[c],
For: Number of Table Entries
Aqueous Saturation[a], Aqueous Relative Permeability[b],
Format: *Real*[a], *Real*[b],
Endfor: Number of Table Entries
Endif:
Elseif: Operational Mode Option: Solution Control Card = { Water-Oil } { Water-Oil-Air } { Water-Oil-Air-Energy } { Water-Oil-Dissolved Oil } { Water-Oil-Dissolved Oil-Surfactant }
Rock/Soil Name[a],
Permeability Function Option[b]
{ Mualem } { Burdine }
If: Saturation Function Option = { van Genuchten }
van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = { Brooks and Corey }
Brooks and Corey "λ" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Endif:
Endif:
Endfor: Number of Rock/Soil Types

Endcard: Aqueous Relative Permeability Card

A.10.1 Aqueous Relative Permeability Card Examples

```
~Aqueous Relative Permeability Card
Sand,Mualem,,
```

```
~Aqueous Relative Permeability Card
20/30 Ottawa Sand,Mualem,0.56,
```

```
~Aqueous Relative Permeability Card
Sand,Fatt And Klikoff,
```

```
~Aqueous Rel
SM-ML1,Burdine,,
SW1,Burdine,,
SP3,Burdine,,
SM-SP1,Burdine,,
SP2,Burdine,,
SP1,Burdine,,
US,Touma and Vauclin,1.0,2.0,
```

A.11 Gas Relative Permeability Card

Card Title[a] { ~Gas Rel [ative Permeability Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

If: Operational Mode: Solution Control Card = { Water-Air } { Water-Air-Energy } { Water-Air-Salt } { Water-Air-Energy-Salt }
Rock/Soil Name[a],
Permeability Function Option[b],
{ Constant } { Mualem } { Burdine } { Fatt and Klikoff } { Corey } { Tabular }
If: Permeability Function Option = { Constant }
If: Rock/Soil Name = {{ Fractured }}
Matrix Gas Relative Permeability[c],
Fracture Gas Relative Permeability[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[d],
Else:
Gas Relative Permeability[c]
Format: *Char*[a], *Char*[b], *Real*[c],
Endif:
Elseif: Permeability Function Option = { Mualem } or { Burdine }
If: Saturation Function Option = {{ van Genuchten }}
van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ Brooks and Corey }}
Brooks and Corey "λ" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ van Genuchten }} and Rock/Soil Name = {{ Fractured }}
Matrix van Genuchten "m" parameter[c],
Fracture van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[a],
Elseif: Saturation Function Option = {{ Brooks and Corey }} and Rock/Soil Name = {{ Fractured }}
Matrix Brooks and Corey "λ" parameter[c],
Fracture Brooks and Corey "λ" parameter[d],
Format: *Char*[a], *Char*[b], *Real*[c], *Real*[a],
Endif:
Elseif: Permeability Function Option = { Tabular }
Number of Table Entries[c],
Format: Char[a],Char[b],Integer[c],
For: Number of Table Entries
Gas Saturation[a], Gas Relative Permeability[b],
Format: *Real*[a], *Real*[b],
Endfor:
Endif:

Gas Relative Permeability Card (contd)

Elseif: Operational Mode: Solution Control Card = { Water-Oil-Air } { Water-Oil-Air-Energy }
Rock/Soil Name[a],
Permeability Function Option[b],
{ Mualem }
{ Burdine }
If: Saturation Function Option = {{ van Genuchten }}
van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ Brooks and Corey }}
Brooks and Corey "λ" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Endif:
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Gas Relative Permeability Card

A.11.1 Gas Relative Permeability Examples

```
~Gas Relative Permeability Card
Sand,Mualem,,
```

```
~Gas Rel
20/30 Ottawa Sand,Mualem,0.56,
```

```
~Gas Relative Permeability Card
Sand,Fatt And Klikoff,
```

```
~Gas Relative Permeability
SM-ML1,Burdine,,
SW1,Burdine,,
# Tabular Input
SP3,Tabular,6,
1.0, 1.0,
0.8, 0.64,
0.6, 0.36,
0.4, 0.16,
0.2, 0.04,
0.0, 0.0,
SM-SP1,Mualem,0.5,
SP2,Burdine,,
SP1,Burdine,,
US,Touma and Vauclin,1.0,2.0,
```

A.12 NAPL Relative Permeability Card

Card Title[a] { ~NAPL Rel [ative Permeability Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

If: Operational Mode: Solution Control Card = { Water-Oil } { Water-Air-Oil } { Water-Air-Oil-Energy } { Water-Oil-Dissolved Oil } { Water-Oil-Dissolved Oil-Surfactant }
Rock/Soil Name[a],
Permeability Function Option[b],
{ Constant } { Mualem } { Burdine }
If: Permeability Function Option = { Constant }
Gas Relative Permeability[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Permeability Function Option = { Mualem } or { Burdine }
If: Saturation Function Option = {{ van Genuchten }}
van Genuchten "m" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Elseif: Saturation Function Option = {{ Brooks and Corey }}
Brooks and Corey "λ" parameter[c],
Format: *Char*[a], *Char*[b], *Real*[c],
Endif:
Endif:
Endif:

Endfor: Number of Rock/Soil Types

Endcard: NAPL Relative Permeability Card

A.12.1 NAPL Relative Permeability Card Examples

```
~NAPL Relative Permeability Card
Sand,Mualem,,
```

```
~NAPL Relative Permeability
20/30 Ottawa Sand,Constant,0.58,
```

```
~NAPL Relative Permeability Input Card
Sand,Mualem,0.56,
```

```
~NAPL Rel
SM-ML1,Burdine,,
# Constant NAPL relative permeability
SW1,Constant,1.0,
SP3,Constant,1.0,
SM-SP1,Burdine,,
SP2,Burdine,,
SP1,Burdine,,
US,Mualem,,
```

A.13 Oil Properties Card

Card Title[a] { ~Volatile Organic Compound Prop [erties Card] }
{ ~Organic Compound Prop [erties Card] }
{ ~VOC Prop [erties Card] } { ~Oil Prop [erties Card] }
Format: *Char*[a]

Oil Name[a],
Format: *Char*[a],

Molecular Weight[a], Units[b] (kgmol/kg),
Freezing Point Temperature[c], Units[d] (K),
Normal Boiling Point[e], Units[f] (K),
Critical Temperature[g], Units[h] (K),
Format: *Real*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Char*[h],

Critical Pressure[a], Units[b] (bar),
Critical Molar Volume[c], Units[d] (cm^3/mole),
Critical Compressibility[e],
Pitzner Acentric Factor[f],
Dipole Moment[g], Units[h] (Debyes),
Format: *Real*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Real*[f], *Real*[g], *Char*[h],

Note: $\bar{c}^{o}_{pn} = a + bT + cT^2 + dT^3, \frac{J}{mol\,K}$

Isobaric Molar Specific Heat Constants a[a],
Isobaric Molar Specific Heat Constants b[b],
Isobaric Molar Specific Heat Constants c[c],
Isobaric Molar Specific Heat Constants d[d],
Format: *Real*[a], *Real*[b], *Real*[c], *Real*[d],

Oil Properties Card (contd)

Saturated Vapor Pressure Function Option[a]

{ Equation 1 } **Note:** $\ln\left(\frac{P_g^o}{P_c^o}\right) = \frac{\left(ax+bx^{1.5}+cx^3+dx^6\right)}{(1-x)}$ *where,* $x = 1-\frac{T}{T_c^o}$

{ Equation 2 } **Note:** $\ln\left(P_g^o\right) = a-\frac{b}{T}+c\ln(T)+d\left(\frac{P_g^o}{T^2}\right)$

{ Equation 3 } **Note:** $\ln\left(P_g^o\right) = a-\frac{b}{(c+T)}$

{ Constant } **Note:** $P_g^o = \overline{P}_g^o$

If: Saturated Vapor Pressure Function Option = { Equation 1 }
Constant a[b], Constant b[c], Constant c[d], Constant d[e],
Format: *Char*[a], *Real*[b], *Real*[c], *Real*[d], *Real*[e],
Elseif: Saturated Vapor Pressure Function Option = { Equation 2 }
Constant a[b], Constant b[c], Constant c[d], Constant d[e],
Format: *Char*[a], *Real*[b], *Real*[c], *Real*[d], *Real*[e],
Elseif: Saturated Vapor Pressure Function Option = { Equation 3 }
Constant a[b], Constant b[c], Constant c[d],
Format: *Char*[a], *Real*[b], *Real*[c], *Real*[d],
Elseif: Saturated Vapor Pressure Function Option = { Constant }
Saturated Oil Vapor Pressure[b], Units[c] (Pa),
Format: *Char*[a], *Real*[b], *Char*[c],
Endif:

Oil Properties Card (contd)

Liquid Density Function Option[a]

{ HBT Technique } **Note:** [Reid et al. 1987, pp. 55-66]

{ Modified Rackett w/Reference } **Note:** [Reid et al. 1987, pp. 67]

{ Modified Rackett } **Note:** [Reid et al. 1987, pp. 67]

{ Constant } **Note:** $\rho_n^o = \overline{\rho}_n^o$

If: Liquid Density Function Option = { HBT Technique }
Pure Component Characteristic Volume[b], Units[c] (L/mol),
HBT Acentric Factor[d],
Format: *Char[a], Real[b], Char[c], Real[d],*
Elseif: Liquid Density Function Option = { Modified Rackett w/ Reference }
Rackett Compressibility Factor[b],
HBT Acentric Factor[c],
Reference Liquid Density[d], Units[e] (kg/m^3),
Reference Temperature[f], Units[g] (C),
Format: *Char[a], Real[b], Real[c], Real[d], Char[e], Real[f], Char[g],*
Elseif: Liquid Density Function Option = { Modified Rackett }
Rackett Compressibility Factor[b],
HBT Acentric Factor[c],
Format: *Char[a],Real[b],Real[c],*
Elseif: Liquid Density Function Option = { Constant }
Reference Liquid Density[b], Units[c] (kg/m^3),
Format: *Char[a],Real[b],Char[c],*
Endif:

Oil Properties Card (contd)

Liquid Viscosity Function Option[a]

{ Reference } **Note:** $\mu_n^o = \left[\left(\overline{\mu}_n^o\right)^{-0.2661} + \frac{\left(T-\overline{T}_n^o\right)}{233.0}\right]^{-3.758}$ *in cP and K*

{ Equation 1 } **Note:** $\mu_n^o = aT^b$ *in cP and K*

{ Equation 2 } **Note:** $\mu_n^o = \exp\left(a+\frac{b}{T}\right)$ *in cP and K*

{ Equation 3 } **Note:** $\mu_n^o = \exp\left(a+\frac{b}{T}+cT+dT^2\right)$ *in cP and K*

{ Constant } **Note:** $\mu_n^o = \overline{\mu}_n^o$

If: Liquid Viscosity Function Option = { Reference }
Reference Liquid Viscosity[b], Units[c] (Pa s),
Reference Temperature[d], Units[e] (C),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e],*
Elseif: Liquid Viscosity Function Option = { Equation 1 }
Constant a[b], Constant b[c],
Format: *Char[a], Real[b], Real[c],*
Elseif: Liquid Viscosity Function Option = { Equation 2 }
Constant a[b], Constant b[c],
Format: *Char[a], Real[b], Real[c],*
Elseif: Liquid Viscosity Function Option = { Equation 3 }
Constant a[b], Constant b[c],
Constant c[d], Constant d[e],
Format: *Char[a], Real[b], Real[c], Real[d], Real[e],*
Elseif: Liquid Viscosity Function Option = { Constant }
Reference Liquid Viscosity[b], Units[c] (Pa s),
Format: *Char[a], Real[b], Char[c],*
Endif:

Henry's Constant for Aqueous Solubility[a] (1.e+20), Units[b] (Pa),
Format: *Real[a],Char[b],*

Endcard: Oil Properties Card

A.13.1 Oil Properties Card Examples

```
~Oil Properties Card
PCE,
165.834,kg/kgmol,251.,K,394.4,K,620.2,K,
47.6,bar,289.6,cm^3/mol,0.2758,0.2515,0.0,debyes,
,,,,
Equation 1,-7.36067,1.82732,-3.47735,-1.00033,
Modified Rackett,0.2758,0.2515,
Equation 2,-3.334,946.4,
9.463e+07,Pa,
```

```
~Volatile Organic Compound Properties Card
NAPL,
170.34,g/mol,-9.55,C,225.35,C,385.05,C,
18.2,bar,713,cm^3/mol,0.24,0.0,0,debyes,
-9.328,1.149,-0.0006347,1.359e-07,
Equation 2,77.628,10012.5,-9.236,10030.0,
Constant,840,kg/m^3,
Constant,0.0047,Pa s,
1.0e10,Pa,
```

A.14 Dissolved-Oil Transport Card

Card Title[a] { ~Dissolved Oil Transport [Card] } {~Dissolved VOC Transport [Card] }
{ ~Dissolved Organic Transport [Card] }
Format: *Char*[a]

If: Operational Mode: Solution Control Card = { Water-Oil-Dissolved Oil }
{ Water-Oil-Dissolved Oil-Surfactant }
Interphase Mass Transfer Function Option[a],

{ Welty } **Note:** $$Sh = \frac{k^o_{n\ell}\, d_p}{D^o_\ell} = a + b\, Re^m\, Sc^n$$

{ Parker } **Note:** $$Sh = \frac{k^o_{n\ell}\, d_p}{D^o_\ell} = a + b\, Re^m \left(s_n\, n_D\right)^n$$

Constant a[b], Constant b[c], Constant m[d], Constant n[e],
Format: *Char*[a], *Real*[b], *Real*[c], *Real*[d], *Real*[e],
Endif:

Dissolved-Oil Transport Card (contd)

For: Number of Rock/Soil Types

If: Operational Mode: Solution Control Card = { Water-Oil } { Water-Air-Oil }
Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m),
Dissolved-Oil Adsorption Function Option[f],

{ Linear } **Note:** $R_D = \left[1+\frac{k}{s_\ell n_D}\right]$

{ Linear Kd } **Note:** $R_D = \left[1+\frac{K_d \rho_b}{s_\ell n_D}\right]$

{ Freundlich } **Note:** $R_D = \left[1+\frac{n k C_\ell^{n-1}}{s_\ell n_D}\right]$

{ Langmuir } **Note:** $R_D = \left[1+\frac{a}{s_\ell n_D\left(1+bC_\ell\right)^2}\right]$

If: Dissolved-Oil Adsorption Function Option = { Linear }
Constant k[g],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g],*
Elseif: Dissolved-Oil Adsorption Function Option = { Linear Kd }
Constant Kd[g], Units[h] (m^3/kg),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h],*
Elseif: Dissolved-Oil Adsorption Function Option = { Freundlich }
Constant k[g],
Constant n[h],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h],*
Elseif: Dissolved-Oil Adsorption Function Option = { Langmuir }
Constant a[g], Constant b[h], Units[i] (m^3/kg),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Char[i],*
Endif:

Endif:

Dissolved-Oil Transport Card (contd)

Elseif: Operational Mode: Solution Control Card = { Water-Oil-Dissolved Oil }
{ Water-Oil-Dissolved Oil-Surfactant }
Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m),
Dissolved-Oil Adsorption Function Option[f],

{ Linear } **Note:** $R_D = \left[1+\frac{k}{s_\ell n_D}\right]$

{ Linear Kd } **Note:** $R_D = \left[1+\frac{K_d \rho_b}{s_\ell n_D}\right]$

{ Freundlich } **Note:** $R_D = \left[1+\frac{n k C_\ell^{n-1}}{s_\ell n_D}\right]$

{ Langmuir } **Note:** $R_D = \left[1+\frac{a}{s_\ell n_D \left(1+bC_\ell\right)^2}\right]$

If: Dissolved-Oil Adsorption Function Option = { Linear }
Constant k[g], Nominal Particle Diameter[h], Units[i] (m),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Char[i],*
Elseif: Dissolved-Oil Adsorption Function Option = { Linear Kd }
Constant Kd[g], Units[h] (m^3/kg), Nominal Particle Diameter[i], Units[j] (m),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i], Char[j],*
Elseif: Dissolved-Oil Adsorption Function Option = { Freundlich }
Constant k[g], Constant n[h], Nominal Particle Diameter[i], Units[j] (m),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Real[h], Char[i],*
Elseif: Dissolved-Oil Adsorption Function Option = { Langmuir }
Constant a[g], Constant b[h], Units[i] (m^3/kg), Nominal Particle Diameter[j], Units[k] (m),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Char[i], Real[j], Char[k],*
Endif:
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Dissolved-Oil Transport Card

A.14.1 Dissolved-Oil Transport Card Examples

```
~Dissolved Oil Transport Card
Welty,0.55,0.25,1.5,1.5,
20/30 Ottawa Sand,0.154,cm,0.0154,cm,Linear Kd,0.,m^3/kg,0.71,mm,
```

A.15 Surfactant Property/Transport Card

Card Title[a] { ~Surfactant [Property/Transport Card] }
Format: *Char*[a]

Surfactant Name[a],
Molecular Weight[b], Units[c] (kg/kgmol),
Critical Temperature[d], Units[e] (K),
Critical Pressure[f], Units[g] (Pa),
Critical Molar Volume[h], Units[i] (cm^3/mole),
Critical Compressibility[j], Pitzner Accentric Factor[k],
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Char*[g], *Real*[h], *Char*[i], *Real*[j], *Real*[k],

Critical Micellar Concentration[a], Units[b] (kg/m^3),
Solubility Function Slope[c],
Oil-Water Interfacial Tension w/ Surfactant[d], Units[e] (N/m),
Format: *Real*[a], *Char*[b], *Real*[c], *Real*[d], *Char*[e],

Surfactant Property/Transport Card (contd)

Liquid Viscosity Function Option[a]

{ Reference } **Note:** $\mu_n^o = \left[\left(\overline{\mu}_n^o\right)^{-0.2661} + \frac{\left(T - \overline{T}_n^o\right)}{233.0}\right]^{-3.758}$ *in cP and K*

{ Equation 1 } **Note:** $\mu_n^o = aT^b$ *in cP and K*

{ Equation 2 } **Note:** $\mu_n^o = \exp\left(a + \frac{b}{T}\right)$ *in cP and K*

{ Equation 3 } **Note:** $\mu_n^o = \exp\left(a + \frac{b}{T} + cT + dT^2\right)$ *in cP and K*

{ Constant } **Note:** $\mu_n^o = \overline{\mu}_n^o$

If: Liquid Viscosity Function Option = { Reference }
Reference Liquid Viscosity[b], Units[c] (Pa s),
Reference Temperature[d], Units[e] (C),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e],*
Elseif: Liquid Viscosity Function Option = { Equation 1 }
Constant a[b],
Constant b[c],
Format: *Char[a], Real[b], Real[c],*
Elseif: Liquid Viscosity Function Option = { Equation 2 }
Constant a[b],
Constant b[c],
Format: *Char[a], Real[b], Real[c],*
Elseif: Liquid Viscosity Function Option = { Equation 3 }
Constant a[b],
Constant b[c],
Constant c[d],
Constant d[e],
Format: *Char[a], Real[b], Real[c], Real[d], Real[e],*
Elseif: Liquid Viscosity Function Option = { Constant }
Reference Liquid Viscosity[b], Units[c] (Pa s),
Format: *Char[a], Real[b], Char[c],*
Endif:

Surfactant Property/Transport Card (contd)

For: Number of Rock/Soil Types

Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m),
Surfactant Adsorption Function Option[f],

{ Linear } **Note:** $R_D = \left[1+\frac{k}{s_\ell n_D}\right]$

{ Linear Kd } **Note:** $R_D = \left[1+\frac{K_d \rho_b}{s_\ell n_D}\right]$

{ Freundlich } **Note:** $R_D = \left[1+\frac{n k C_\ell^{n-1}}{s_\ell n_D}\right]$

{ Langmuir } **Note:** $R_D = \left[1+\frac{a}{s_\ell n_D (1+bC_\ell)^2}\right]$

If: Surfactant Adsorption Function Option = { Linear }
Constant k[g],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g],*
Elseif: Surfactant Adsorption Function Option = { Linear Kd }
Constant Kd[g], Units[h] (m^3/kg),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h],*
Elseif: Surfactant Adsorption Function Option = { Freundlich }
Constant k[g],
Constant n[h],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h],*
Elseif: Surfactant Adsorption Function Option = { Langmuir }
Constant a[g],
Constant b[h], Units[i] (m^3/kg),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Char[i],*
Endif:

Endfor: Number of Rock/Soil Types

Endcard: Surfactant Property/Transport Card

A.15.1 Surfactant Property Transport Card Examples

```
~Surfactant Property/Transport Card
Witconol 2722,1309.67,kg/kgmol,750.528,K,5.165e+5,Pa, ¬
   3865.5,cm^3/mol,0.29015,0.0047,
# Surfactant similar to T-MAZ-80.
13.0,g/m^3,0.8471,0.09,dynes/cm,
Reference,180,cP,23,C,
20/30 Ottawa Sand,0.154,cm,0.0154,cm,Linear Kd,0.,m^3/kg,
```

A.16 Salt Transport Card

Card Title[a] { ~Salt Transport [Card] }
Format: *Char*[a]

If: Operational Mode Option: Solution Control Card = { Water-Salt } { Water-Air-Salt}
{ Water-Air-Energy-Salt }
Effective Diffusion Option[a],

{ Conventional } **Note:** $D^s_{\ell e} = \tau_\ell \, s_\ell \, n_D \, D^s_\ell$

{ Empirical [Kemper and van Schaik] } **Note:** $D^s_{\ell e} = D^s_\ell \, a \exp\left(b \, n_D \, s_\ell\right)$

If: Effective Diffusion Option = { Conventional }
Aqueous-Phase Diffusion Coefficient @ 20 C[b], Units[c] (m^2/s),
Format: *Char*[a], *Real*[b], *Char*[c],
Elseif: Effective Diffusion Option = { Empirical [Kemper and van Schaik] }
Format: *Char*[a],
Endif:
For: Number of Rock/Soil Types
If: Effective Diffusion Option = { Conventional }
Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m),
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e],
Elseif: Effective Diffusion Option = { Empirical }
Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m),
Aqueous Molecular Diffusion Coefficient[f], Units[g] (m^2/s),
Constant a[h],
Constant b[i],
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Char*[g], *Real*[h], *Real*[i],
Endif:
Endfor: Number of Rock/Soil Types
Endif:

Endcard: Salt Transport Card

A.16.1 Salt Transport Card Examples

```
~Salt Transport Card
Constant Diffusion,3.565e-6,m^2/s,
Geologic Media,0.0,m,0.0,m,
```

```
~Salt Transport Card
Conventional,1.157e-10,m^2/s,
Backfill,1.0,m,0.1,m,
Hanford Fine,1.0,m,0.1,m,
Plio-Plesitocene,1.0,m,0.1,m,
Ringold Gravel,1.0,m,0.1,m,
```

A.17 Solute/Fluid Interactions Card

Card Title[a] { ~Solute/Fluid [Interactions Card] }
Format: *Char*[a]

Number of Solutes
Format: *Integer,*

For: Number of Solutes

If: Operational Mode Option: Solution Control Card = { Water } { Water-Salt }
Solute Name[a],
Effective Diffusion Option[b],

{ Constant } **Note:** $D_{\ell\,e}^{C} = \overline{D}_{\ell\,e}^{C}$

{ Conventional } **Note:** $D_{\ell\,e}^{C} = \tau_{\ell}\, s_{\ell}\, n_{D}\, D_{\ell}^{C}$

{ Empirical [Kemper and van Schaik] } **Note:** $D_{\ell\,e}^{C} = D_{\ell}^{C}\, a \exp\left(b\, n_{D}\, s_{\ell}\right)$

If: Effective Diffusion Option = { Empirical }
Solute Partition Option[c],

{ Continuous } **Note:** $C_{\ell} = \dfrac{C}{n_{D}\, s_{\ell} + \left(1 - n_{T}\right)\rho_{s}\, K_{s\ell}}$

{ Noncontinous } **Note:** $C_{\ell} = \dfrac{C}{n_{D}\, s_{\ell} + s_{\ell}\left(1 - n_{T}\right)\rho_{s}\, K_{s\ell}}$

IfDef: Radioactive
Half-Life[d], Units[e] (s),
Format: *Char*[a], *Char*[b], *Char*[c], *Real*[d], *Char*[f],
ElseifDef: Reactive
Number of Reactions[d],
For: Number of Reactions:
First-Order Decay Constant[e], Units[f] (s),
Endfor: Number of Reactions
Format: *Char*[a], *Char*[b], *Char*[c], *Integer*[d],<*Real*[e], *Char*[f],>
EndifDef:
Else:
Aqueous-Phase Molecular Diffusion Coefficient @ 20 C[c], Units[d] (m^2/s),
Solute Partition Option[e],

{ Continuous } **Note:** $C_{\ell} = \dfrac{C}{n_{D}\, s_{\ell} + \left(1 - n_{T}\right)\rho_{s}\, K_{s\ell}}$

{ Noncontinous } **Note:** $C_{\ell} = \dfrac{C}{n_{D}\, s_{\ell} + s_{\ell}\left(1 - n_{T}\right)\rho_{s}\, K_{s\ell}}$

Solute/Fluid Interactions Card (contd)

IfDef: Radioactive
Half-Life[f], Units[g] (s),
Format: *Char[a], Char[b], Real[c], Char[d], Char[e], Real[f], Char[g],*
ElseifDef: Reactive
Number of Reactions[f],
For: Number of Reactions:
First-Order Decay Constant[g], Units[h] (s),
Endfor: Number of Reactions
Format: *Char[a], Char[b], Real[c], Char[d], Char[e], Integer[f], <Real[g], Char[h],>*
EndifDef:
Endif:

Elseif: Operational Mode Option: Solution Control Card = { Water-Air } { Water-Air-Energy } { Water-Air-Salt } { Water-Air-Energy-Salt }
Solute Name[a],
Aqueous-Phase Molecular Diffusion Coefficient @ 20 C[b], Units[c] (m^2/s),
Gas-Phase Molecular Diffusion Coefficient @ 20 C[d], Units[e] (m^2/s),
Gas-Aqueous Partition Function Option[f]

{ Constant } **Note:** $K_{g\ell} = \overline{K}_{gl}$

{ Temperature Dependent } **Note:** $\ln\left(K_{g\ell}\right) = a + \frac{b}{T} + c\ln\left(T\right) + d\,T + e\,T^2$

If: Gas-Aqueous Partition Function Option = { Constant }
Gas-Aqueous Partition Coefficient[g], Units[h] (m^3/m^3),
IfDef: Radioactive
Half-Life[i], Units[j] (s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i], Char[j],*
ElseifDef: Reactive
Number of Parent Reactions[i],
For: Number of Reactions
First-Order Decay Constant[j], Units[k] (s),
Endfor: Number of Reactions
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Integer[i],<Real[j], Char[k],>*
EndifDef:

Elseif: Gas-Aqueous Partition Function Option = { Temperature Dependent }
Constant a[g], Constant b[h], Constant c[i], Constant d[j], Constant e[k],
IfDef: Radioactive
Half-Life[l], Units[m] (s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Real[i], Real[j], Real[k], Real[l], Char[m],*
ElseifDef: Reactive
Number of Parent Reactions[l],
For: Number of Reactions
First-Order Decay Constant[m], Units[n] (s),
Endfor: Number of Reactions
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Real[h], Real[i], Real[j], Real[k], Integer[l],<Real[m], Char[n],>*
EndifDef:
Endif:

Solute/Fluid Interactions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
{ Water-Oil-Dissolved Oil } { Water-Oil-Dissolved Oil-Surfactant }
Solute Name[a], Aqueous-Phase Molecular Diffusion Coefficient[b], Units[c] (m^2/s),
NAPL Molecular Diffusion Coefficient[d] @ 20 C, Units[e] (m^2/s),
Aqueous-NAPL Partition Coefficient[f], Units[g] (m^3/m^3),
IfDef: Radioactive
Half-Life[h], Units[i] (s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Real[f], Char[g], Real[h], Char[i],*
ElseifDef: Reactive
Number of Parent Reactions[h],
For: Number of Reactions
First-Order Decay Constant[i], Units[j] (s),
Endfor: Number of Reactions
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Real[f], Char[g],*
Integer[h], <Real[i], Char[j],>
EndifDef:

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
or { Water-Oil-Air-Energy }
Solute Name[a],
Aqueous-Phase Molecular Diffusion Coefficient[b] @ 20 C, Units[c] (m^2/s),
Gas-Phase Molecular Diffusion Coefficient[d] @ 20 C, Units[e] (m^2/s),
NAPL Molecular Diffusion Coefficient[f] @ 20 C, Units[g] (m^2/s),
Gas-Aqueous Partition Function Option[h]

{ Constant } **Note:** $K_{g\ell} = \overline{K}_{gl}$

{ Temperature Dependent } **Note:** $\ln\left(K_{g\ell}\right) = a + \frac{b}{T} + c\ln\left(T\right) + d\,T + e\,T^2$

If: Gas-Aqueous Partition Function Option = { Constant }
Gas-Aqueous Partition Coefficient[i], Units[j] (m^3/m^3),
Aqueous-NAPL Partition Coefficient[k], Units[l] (m^3/m^3),
IfDef: Radioactive
Half-Life[m], Units[n] (s),
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i],*
Char[j], Real[k], Char[l], Real[m], Char[n],
ElseifDef: Reactive
Number of Parent Reactions[m],
For: Number of Reactions
First-Order Decay Constant[n], Units[o] (s),
Endfor: Number of Reactions
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i],*
Char[j], Real[k], Char[l], Integer[m], <Real[n], Char[o],>
EndifDef:

Solute/Fluid Interactions Card (contd)

Elseif: Gas-Aqueous Partition Function Option = { Temperature Dependent }
Constant a[i], Constant b[j], Constant c[k], Constant d[l], Constant e[m],
Aqueous-NAPL Partition Coefficient[n], Units[o] (m^3/m^3),
IfDef: Radioactive
Half-Life[p], Units[q] (s)
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i], Real[j], Real[k], Real[l], Real[m], Real[n], Char[o], Real[p], Char[q],*
ElseifDef: Reactive
Number of Parent Reactions[p],
For: Number of Reactions
First-Order Decay Constant[q], Units[r] (s),
Endfor: Number of Reactions
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Char[f], Real[g], Char[h], Real[i], Real[j], Real[k], Real[l], Real[m], Real[n], Char[o], Integer[p], <Real[q], Char[r],>*
EndifDef:
Endif:
Endif:

Endfor: Number of Rock/Soil Types

IfDef: Radioactive
Number of Chain Decay Lines[a],
Format: *Integer[a],*
For: Number of Chain Decay Lines
Parent Solute Name[a], Progeny Solute Name[b], Chain Decay Fraction[c],
Format: *Char[a], Char[b], Real[c],*
Endfor: Number of Chain Decay Lines
ElseifDef: Reactive
Number of Reaction Lines[a],
Format: *Integer[a],*
For: Number of Reaction Lines
Parent Solute Name[a], Progeny Solute Name[b], Reaction Stoichiometry Fraction[c],
Format: *Char[a], Char[b], Real[c],*
Endfor: Number of Reaction Lines
EndifDef

Endcard: Solute/Fluid Interactions Card

A.17.1 Solute/Fluid Interaction Card Examples

```
~Solute/Fluid Interaction Card
1,
TCE,Conventional,9.6283e-6,cm^2/s,Continuous,1000,yr,
0,
```

```
~Solute/Fluid Interaction Card
4,
Tc-99,Empirical,Noncontinuous,1.e20,yr,
U-238,Empirical,Noncontinuous,1.e20,yr,
Np-237,Empirical,Noncontinuous,1.e20,yr,
Pu-239,Empirical,Noncontinuous,1.e20,yr,
0,
```

```
~Solute/Fluid Interaction Card
7,
Dithionite,Conventional,1.e-5,cm^2/s,Continuous,2,5,hr,18,hr,
Sulfite,Conventional,1.e-5,cm^2/s,Continuous,0,
Bisulfite,Conventional,1.e-5,cm^2/s,Continuous,0,
Thiosulfate,Conventional,1.e-5,cm^2/s,Continuous,0,
Fe(III),Conventional,1.e-5,cm^2/s,Continuous,0,
Fe(II),Conventional,1.e-5,cm^2/s,Continuous,0,
H+,Conventional,1.e-5,cm^2/s,Continuous,0,
6,
Dithionite,Fe(III),1,-2.0,
Dithionite,Fe(II),1,2.0,
Dithionite,Sulfite,1,2.0,
Dithionite,H+,1,4.0,
Dithionite,Bisulfite,2,1.0,
Dithionite,Thiosulfate,2,0.5,
```

A.18 Solute/Porous Media Interactions Card

Card Title[a] { ~Solute/Porous [Media Inteactions Card] }
Format: *Char*[a]

For: Number of Rock/Soil Types

Rock/Soil Name[a],
Longitudinal Dispersivity[b], Units[c] (m),
Transverse Dispersivity[d], Units[e] (m)
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e],

For: Number of Solutes
Solute Name[a],
Solid-Aqueous Partition Coefficient[b], Units[c] (m^3/kg),
If: Effective Diffusion Option: Solute/Fluid Interactions Card = { Empirical }
Aqueous Molecular Diffusion Coefficient[d], Units[e] (m^2/s),
Solute/Fluid Interactions Card: Constant a[f],
Solute/Fluid Interactions Card: Constant b[g],
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Real*[g],
Else:
Format: *Char*[a], *Real*[b], *Char*[c],
Endif:
Endfor: Number of Solutes

Endfor: Number of Rock/Soil Types

Endcard: Solute/Porous Media Interactions Card

A.18.1 Solute/Porous Media Interactions Card Examples

```
~Solute/Porous Media Interaction Card
Backfill Soil,,,,,
Tc-99,0.,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
U-238,0.67,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Np-237,3.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Pu-239,21.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Hanford Sand,,,,,
Tc-99,0.,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
U-238,0.67,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Np-237,3.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Pu-239,21.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Hanford Gravel,,,,,
Tc-99,0.,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
U-238,0.67,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Np-237,3.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Pu-239,21.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Ringold Formation,,,,,
Tc-99,0.,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
U-238,0.67,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Np-237,3.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Pu-239,21.0,cm^3/g,7.8894e-2,m^2/yr,0.005,10.,
Glass Waste,,,,,
Tc-99,0.,cm^3/g,3.1557e-7,m^2/yr,1.,0.,
U-238,0.,cm^3/g,3.1557e-7,m^2/yr,1.,0.,
Np-237,0.,cm^3/g,3.1557e-7,m^2/yr,1.,0.,
Pu-239,0.,cm^3/g,3.1557e-7,m^2/yr,1.,0.,
```

```
~Solute/Porous Media Interaction Card
Soil,1,m,0.1,m,
Dithionite,0.0,m^3/kg,
Sulfite,0.0,m^3/kg,
Bisulfite,0.0,m^3/kg,
Thiosulfate,0.0,m^3/kg,
Fe(III),0.0,m^3/kg,
Fe(II),0.0,m^3/kg,
H+,0.0,m^3/kg,
```

A.19 Initial Conditions Card

Card Title[a] { ~Initial [Conditions Card] }
Format: *Char*[a]

If: Operational Mode Option: Solution Control Card = { Water } { Water-Air }
{ Water-Air-Energy w/o Ice } { Water-Salt } { Water-Air-Salt } { Water-Air-Energy-Salt }
Initial Saturation Option[a], Initial Saturation Option[b],
{ Gas Pressure, Aqueous Pressure }
{ Gas Pressure, Aqueous Saturation }
{ Aqueous Pressure, Aqueous Saturation }
Format: *Char*[a], *Char*[b],

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy w/ Ice }
Initial Saturation Option[a], Initial Saturation Option[b], Initial Saturation Option[c],
{ Gas Pressure, Aqueous Pressure, Temperature }
{ Gas Pressure, Aqueous Pressure, Ice Saturation }
{ Gas Pressure, Aqueous Saturation, Ice Saturation }
{ Gas Pressure, Apparent Aqueous Saturation, Temperature }
Format: *Char*[a], *Char*[b], *Char*[c],
Endif:

Initial Conditions Card (contd)

Number of Initial Conditions Domains[a]
Format: *Integer*[a],

For: Number of Initial Conditions Domains

If: Operational Mode Option: Solution Control Card = { Water }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] }
{ Aqueous Saturation } { Trapped Gas Saturation [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] }
{ Aqueous Saturation } { Trapped Gas Saturation [Overwrite] }
{ Salt [Volumetric Conc.] [Overwrite] } { Salt Aqueous [Volumetric Conc.] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
or { Water-Air-Energy w/o Ice }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] }
{ Aqueous Saturation } { Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Air Mole Fraction [Overwrite] }
{ Aqueous Dissolved Air Mass Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute Gas [Volumetric Conc.], Solute Name }

Initial Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Salt }
or { Water-Air-Energy-Salt }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] }
{ Aqueous Saturation }{ Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Air Mole Fraction [Overwrite] }
{ Aqueous Dissolved Air Mass Fraction [Overwrite] }
{ Salt [Volumetric Conc.] [Overwrite] } { Salt Aqueous [Volumetric Conc.] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute Gas [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy w/ Ice }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { Aqueous Saturation }
{ Ice Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Apparent Aqueous Saturation }
{ Aqueous Dissolved Air Mole Fraction [Overwrite] }
{ Aqueous Dissolved Air Mass Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute Gas [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy-Salt w/ Ice }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { Aqueous Saturation }
{ Ice Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Apparent Aqueous Saturation }
{ Aqueous Dissolved Air Mole Fraction [Overwrite] }
{ Aqueous Dissolved Air Mass Fraction [Overwrite] }
{ Salt [Volumetric Conc.] [Overwrite] } { Salt Aqueous [Volumetric Conc.] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute Gas [Volumetric Conc.], Solute Name }

Initial Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { NAPL Pressure [Overwrite] }
{ Trapped NAPL Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Oil Mole Fraction [Overwrite] }
{ Aqueous Dissolved Oil Mass Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute NAPL [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { NAPL Pressure [Overwrite] }
{ Trapped NAPL Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Oil Mass Fraction [Overwrite] }
{ Aqueous Dissolved Oil Mole Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute NAPL [Volumetric Conc.], Solute Name }

Elseif: Operational Mode Option: Solution Control Card =
{ Water-Oil-Dissolved Oil-Surfactant }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { NAPL Pressure [Overwrite] }
{ Trapped NAPL Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Oil Mass Fraction [Overwrite] }
{ Aqueous Dissolved Oil Mole Fraction [Overwrite] }
{ Aqueous Dissolved Surfactant Mass Fraction [Overwrite] }
{ Aqueous Dissolved Surfactant Mole Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute NAPL [Volumetric Conc.], Solute Name }

Initial Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
or { Water-Oil-Air-Energy }
Variable Name Option[a],
{ Temperature [Overwrite] } { Aqueous Pressure [Overwrite] }
{ Gas Pressure [Overwrite] } { NAPL Pressure [Overwrite] }
{ Trapped NAPL Saturation [Overwrite] } { Trapped Gas Saturation [Overwrite] }
{ Aqueous Dissolved Air Mole Fraction [Overwrite] }
{ Aqueous Dissolved Air Mass Fraction [Overwrite] }
{ Aqueous Dissolved Oil Mole Fraction [Overwrite] }
{ Aqueous Dissolved Oil Mass Fraction [Overwrite] }
{ Solute [Volumetric Conc.] [Overwrite], Solute Name }
{ Solute Aqueous [Volumetric Conc.], Solute Name }
{ Solute Gas [Volumetric Conc.], Solute Name }
{ Solute NAPL [Volumetric Conc.], Solute Name }
Endif:

If: Variable Name Option = {{ Pressure }}
Pressure[b], Units[c] (Pa),
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Elseif: Variable Name Option = { Temperature }
Temperature[b], Units[c] (C),
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b], Char[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Initial Conditions Card (contd)

Elseif: Variable Name Option = {{ Saturation }}
Saturation[b],Null[c],
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b],Null[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Elseif: Variable Name Option = {{ Mass Fraction }}
Mass Fraction[b],Null[c],
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b],Null[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Elseif: Variable Name Option = {{ Mole Fraction }}
Mole Fraction[b],Null[c],
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b],Null[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Initial Conditions Card (contd)

Elseif: Variable Name Option = {{ Salt }}
Volumetric Concentration[b], Units[c] (1/m^3),
X-Dir. Gradient[d], Units[e] (1/m),
Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Real[b],Null[c], Real[d], Char[e], Real[f], Char[g], Real[h],*
Char[i], Integer[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o],

Elseif: Variable Name Option = {{ Solute }}
Solute Name[b], Volumetric Concentration[c], Units[d] (1/m^3),
X-Dir. Gradient[d], Units[e] (1/m), Y-Dir. Gradient[f], Units[g] (1/m),
Z-Dir. Gradient[h], Units[i] (1/m),
I-Start Index[j], I-End Index[k],
J-Start Index[l], J-End Index[m],
K-Start Index[n], K-End Index[o],
Format: *Char[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h],*
Real[i], Char[j], Integer[k], Integer[l], Integer[m], Integer[n], Integer[o], Integer[p],
Endif:

Endfor: Number of Initial Conditions Domains

Endcard: Initial Conditions Card

A.19.1 Initial Conditions Card Examples

```
~Initial Conditions Card
Aqueous Pressure,Gas Pressure,
5,
Temperature,20,C,0,1/m,0,1/m,0,1/m,1,52,1,1,1,113,
Aqueous Pressure,183254,Pa,0,1/m,0,1/m,-9793.52,1/m,1,52,1,1,1,113,
Aqueous Pressure,92956,Pa,0,1/m,0,1/m,0,1/m,2,2,1,1,71,100,
Gas Pressure,92956,Pa,0,1/m,0,1/m,0,1/m,1,52,1,1,1,113,
Solute Aqueous Conc,TCE,1.0,1/ft^3,,,,,,,1,52,1,1,1,61,
```

```
~Initial Conditions Card
5,
Temperature,22.0,C,,,,,,,1,1,1,1,1,4,
Aqueous Pressure,111113.83,Pa,,,,,-9788.83,1/m,1,1,1,1,1,92,
NAPL Pressure,101325.0,Pa,,,,,,,1,1,1,1,1,92,
Gas Pressure,101325.0,Pa,,,,,,,1,1,1,1,1,92,
Trapped Oil,0.11,,,,,,,,1,1,1,1,1,92,
```

```
~Initial Conditions Card
Gas Pressure, Aqueous Pressure,
1,
Aqueous Pressure,121325.,Pa,,,,,-9793.5331,1/m,1,20,1,1,1,10,
```

```
~Initial Conditions Card
Aqueous Saturation,Gas Pressure,
3,
Aqueous Saturation,0.5,,,,,,,,1,50,1,1,1,1,
Gas Pressure,101330,Pa,,,,,,,1,50,1,1,1,1,
Temperature,70.0,C,,,,,,,1,50,1,1,1,1,
```

```
~Initial Conditions Card
2,
Aqueous Pressure,103528.57,Pa,,,,,-9793.5,1/m,1,20,1,1,1,10,
NAPL Pressure,104128.57,Pa,,,,,-9793.5,1/m,1,20,1,1,1,10,
```

A.20 Boundary Conditions Card

Card Title[a] {~Boundary [Conditions Card] }
Format: *Char*[a]

Number of Boundary Condition Domains[a],
Format: *Integer*[a],

For: Number of Boundary Condition Domains

Boundary Surface Direction Option[a],
{ Bottom } { South } { West } { East } { North } { Top }

If: Operational Mode Option: Solution Control Card = { Water }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux } { Saturated } { Unit Gradient }
{ Hydraulic Gradient } { Initial Condition } { Seepage Face }
For: Number of Solutes
Solute Transport Boundary Type Option[c],
{ Volumetric Conc. } { Aqueous Conc. } { Zero Flux }
{ Outflow } { Initial Condition } { Inflow Volumetric } { Inflow Aqueous }
Endfor: Number of Solutes
Format: *Char*[a], *Char*[b], <*Char*[c],>

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux } { Saturated } { Unit Gradient }
{ Hydraulic Gradient } { Seepage Face } { Initial Condition }
Salt Boundary Type Option[c],
{ Volumetric Conc. } { Aqueous Conc. } { Zero Flux }
{ Outflow } { Initial Condition } { Inflow Volumetric } { Inflow Aqueous }
For: Number of Solutes
Solute Transport Boundary Type Option[d],
{ Volumetric Conc. } { Aqueous Conc. } { Zero Flux }
{ Outflow } { Initial Condition } { Inflow Volumetric } { Inflow Aqueous }
Endfor: Number of Solutes
Format: *Char*[a], *Char*[b], *Char*[c], <*Char*[d],>

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux } { Saturated } { Unit Gradient }
{ Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux } { Unit Gradient }
{ Hydraulic Gradient } { Initial Condition }
For: Number of Solutes
Solute Transport Boundary Type Option[d],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow Gas }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], <Char[d],>*

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Salt }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux } { Unit Gradient }
{ Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux } { Unit Gradient }
{ Hydraulic Gradient } { Initial Condition }
Salt Boundary Type Option[d],
{ Volumetric Conc. } { Aqueous Conc. } { Zero Flux }
{ Outflow } { Initial Condition } { Inflow Volumetric } { Inflow Aqueous }
For: Number of Solutes
Solute Transport Boundary Type Option[e],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow Gas }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], <Char[e],>*

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy [w/ Ice] }
Energy Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Outflow } { Initial Condition }
Aqueous-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[d],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
For: Number of Solutes
Solute Transport Boundary Type Option[e],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow Gas }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], <Char[e],>*

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy-Salt [w/Ice] }
Energy Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Outflow } { Initial Condition }
Aqueous-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[d],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Salt Boundary Type Option[e],
{ Volumetric Conc. } { Aqueous Conc. } { Zero Flux }
{ Outflow } { Initial Condition } { Inflow Volumetric } { Inflow Aqueous }
For: Number of Solutes
Solute Transport Boundary Type Option[f],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow Gas }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], Char[e], <Char[f],>*

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
NAPL Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
For: Number of Solutes
Solute Transport Boundary Type Option[d],
{ Volumetric Conc. } { Aqueous Conc. } { NAPL Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow NAPL }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], <Char[d],>*

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
NAPL Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Dissolved Oil Boundary Type Option[d],
{ Aqueous Concentration } { Zero Flux } { Outflow } { Initial Condition }
{ Inflow Aqueous }
For: Number of Solutes
Solute Transport Boundary Type Option[e],
{ Volumetric Conc. } { Aqueous Conc. } { NAPL Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow NAPL }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], <Char[e],>*

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card =
{ Water-Oil-Dissolved Oil-Surfactant }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
NAPL Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Dissolved Oil Boundary Type Option[d],
{ Aqueous Concentration } { Zero Flux } { Outflow } { Initial Condition }
{ Inflow Aqueous }
Surfactant Boundary Type Option[e],
{ Aqueous Concentration } { Zero Flux } { Outflow } { Initial Condition }
{ Inflow Aqueous }
For: Number of Solutes
Solute Transport Boundary Type Option[f],
{ Volumetric Conc. } { Aqueous Conc. } { NAPL Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Volumetric } { Inflow Aqueous } { Inflow NAPL }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], Char[e], <Char[f],>*

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
Aqueous-Phase Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
NAPL Boundary Type Option[d],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
For: Number of Solutes
Solute Transport Boundary Type Option[e],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. } { NAPL Conc. }
{ Zero Flux } { Outflow } { Initial Condition } { Inflow Volumetric }
{ Inflow Aqueous } { Inflow Gas } { Inflow NAPL }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], <Char[e],>*

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
Energy Boundary Type Option[b],
{ Dirichlet } { Neumann } { Zero Flux }
{ Outflow } { Initial Condition }
Aqueous-Phase Boundary Type Option[c],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
Gas-Phase Boundary Type Option[d],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
NAPL Boundary Type Option[e],
{ Dirichlet } { Neumann } { Zero Flux }
{ Unit Gradient } { Hydraulic Gradient } { Initial Condition }
For: Number of Solutes
Solute Transport Boundary Type Option[f],
{ Volumetric Conc. } { Aqueous Conc. } { Gas Conc. } { NAPL Conc. }
{ Zero Flux } { Outflow } { Initial Condition }
{ Inflow Aqueous } { Inflow Gas } { Inflow NAPL }
Endfor: Number of Solutes
Format: *Char[a], Char[b], Char[c], Char[d], Char[e], <Char[f],>*
Endif:

I-Start Index[a], I-End Index[b], J-Start Index[c], J-End Index[d],
K-Start Index[e], K-End Index[f], Number of Boundary Times[g],
Format: *Integer[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g],*

Boundary Conditions Card (contd)

If: Operational Mode Option: Solution Control Card = { Water }
  For: Number of Boundary Times
    Boundary Time[a], Units[b] (s),
    If: Aqueous Boundary Type Option = { Dirichlet } { Zero Flux }
      Aqueous Pressure[c], Units[d] (Pa),
    Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
      Aqueous Volumetric Flux[c], Units[d] (m/s),
    Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
      { Seepage Face }
      Base Aqueous Pressure[c], Units[d] (Pa),
    Else:
      Null[c], Null[d],
    Endif:
    For: Number of Solutes
      If: Solute Transport Boundary Type Option = { Volumetric Conc. }
        Solute Volumetric Concentration[e], Units[f] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
        Solute Aqueous-Phase Volumetric Concentration[e], Units[f] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
        Solute Volumetric Concentration[e], Units[f] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
        Solute Aqueous-Phase Volumetric Concentration[e], Units[f] (1/m^3),
      Else:
        Null[e], Null[f],
      Endif:
    Endfor: Number of Solutes
    Format: *Real[a], Char[b], Real[c], Char[d], [Real[e],] < Real[e], Char[f], >*
  Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
  For: Number of Boundary Times
  Boundary Time[a], Units[b] (s),
    If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
      Aqueous Pressure[c], Units[d] (Pa),
    Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
      Aqueous Volumetric Flux[c], Units[d] (m/s),
    Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
      { Seepage Face }
      Base Aqueous Pressure[c], Units[d] (Pa),
    Else:
      Null[c], Null[d],
    Endif:
    If: Salt Boundary Type Option = { Volumetric Conc. }
      Salt Volumetric Concentration[e], Units[f] (kg/m^3),
    Elseif: Salt Boundary Type Option = { Aqueous Conc. }
      Salt Aqueous-Phase Volumetric Concentration[e], Units[f] (kg/m^3),
    Elseif: Salt Boundary Type Option = { Inflow Volumetric }
      Salt Volumetric Concentration[e], Units[f] (kg/m^3),
    Elseif: Salt Boundary Type Option = { Inflow Aqueous }
      Salt Aqueous-Phase Volumetric Concentration[e], Units[f] (kg/m^3),
    Else:
      Null[e], Null[f],
    Endif:
    For: Number of Solutes
      If: Solute Transport Boundary Type Option = { Volumetric Conc. }
        Solute Volumetric Concentration[g], Units[h] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
        Solute Aqueous-Phase Volumetric Concentration[g], Units[h] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
        Solute Volumetric Concentration[g], Units[h] (1/m^3),
      Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
        Solute Aqueous-Phase Volumetric Concentration[g], Units[h] (1/m^3),
      Else:
        Null[g], Null[h],
      Endif:
    Endfor: Number of Solutes
    Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], <Real[g], Char[h],>*
  Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else:
Null[c], Null[d],
Endif:
Aqueous Dissolved Air Mass Fraction[e],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[f], Units[g] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[f], Units[g] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Gas Aqueous Pressure[f], Units[g] (Pa),
Else:
Null[f], Null[g],
Endif:
Water Vapor Relative Humidity[h],
For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[i], Unitl[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[i], Units[j] (1/m^3),
Else:
Null[i], Null[j],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Real[f], Char[g], Real[h], < Real[i], Char[j], >*
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Salt }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else:
Null[c], Null[d],
Endif:
Aqueous Dissolved Air Mass Fraction[e],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[f], Units[g] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[f], Units[g] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Base Gas Pressure[f], Units[g] (Pa),
Else:
Null[f], Null[g],
Endif:
Water Vapor Relative Humidity[h],
If: Salt Boundary Type Option = { Volumetric Conc. }
Salt Volumetric Concentration[i], Units[j] (kg/m^3),
Elseif: Salt Boundary Type Option = { Aqueous Conc. }
Salt Aqueous-Phase Volumetric Concentration[i], Units[j] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Volumetric }
Salt Volumetric Concentration[i], Units[j] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Aqueous }
Salt Aqueous-Phase Volumetric Concentration[i], Units[j] (kg/m^3),
Else:
Null[i], Null[j],
Endif:
For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Gas Conc. }
Solute Gas Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Else:
Null[k], Null[l],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Real[f], Char[g], Real[h], Real[i], Char[j],*
< Real[k], Char[l], >
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Energy Boundary Type Option = { Dirichlet }
Temperature[c], Units[d] (C),
Elseif: Energy Boundary Type Option = { Neumann }
Energy Flux[c], Units[d] (W/m^2),
Else:
Null[c], Null[d],
Endif:
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[e], Units[f] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[e], Units[f] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[e], Units[f] (Pa),
Else:
Null[e], Null[f],
Endif:
Aqueous Dissolved Air Mass Fraction[g],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[h], Units[i] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[h], Units[i] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Base Gas Pressure[i], Units[i] (Pa),
Else:
Null[h], Null[i],
Endif:
Water Vapor Relative Humidity[j],
For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Gas Conc. }
Solute Gas Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Else:
Null[k], Null[l],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Real[h], Char[i], Real[j], < Real[k], Char[l], >*
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy-Salt }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Energy Boundary Type Option = { Dirichlet }
Temperature[c], Units[d] (C),
Elseif: Energy Boundary Type Option = { Neumann }
Energy Flux[c], Units[d] (W/m^2),
Else:
Null[c], Null[d],
Endif:
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[e], Units[f] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[e], Units[f] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[e], Units[f] (Pa),
Else:
Null[e], Null[f],
Endif:
Aqueous Dissolved Air Mass Fraction[g],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[h], Units[i] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[h], Units[i] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Base Gas Pressure[h], Units[i] (Pa),
Else:
Null[h], Null[i],
Endif:
Water Vapor Relative Humidity[j]
If: Salt Boundary Type Option = { Volumetric Conc. }
Salt Volumetric Concentration[k], Units[l] (kg/m^3),
Elseif: Salt Boundary Type Option = { Aqueous Conc. }
Salt Aqueous-Phase Volumetric Concentration[k], Units[l] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Volumetric }
Salt Volumetric Concentration[k], Units[l] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Aqueous }
Salt Aqueous-Phase Volumetric Concentration[k], Units[l] (kg/m^3),
Else:
Null[k], Null[l],
Endif:

Boundary Conditions Card (contd)

For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Gas Conc. }
Solute Gas Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Else:
Null[m], Null[n],
Endif:
Endfor: Number of Solutes
Format: *Real*[a], *Char*[b], *Real*[c], *Char*[d], *Real*[e], *Char*[f], *Real*[g], *Real*[h], *Char*[i], *Real*[j], *Real*[k], *Char*[l], < *Real*[m], *Char*[n], >
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type ={ Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else:
Null[c], Null[d],
Endif:
Aqueous Dissolved Oil Mass Fraction[e],
If: NAPL Boundary Type Option = { Dirichlet } { Zero Flux }
NAPL Pressure[f], Units[g] (Pa),
Elseif: NAPL Boundary Type Option = { Neumann }
NAPL Volumetric Flux[f], Units[g] (m/s),
Elseif: NAPL Boundary Type Option = { Hydraulic Gradient }
Base NAPL Pressure[f], Units[g] (Pa),
Else:
Null[f], Null[g],
Endif:
For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[h], Units[i] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[h], Units[i] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { NAPL Conc. }
Solute NAPL Volumetric Concentration[h], Units[i] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[h], Units[h] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[h], Units[i] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow NAPL }
Solute NAPL Volumetric Concentration[h], Units[i] (1/m^3),
Else:
Null[h], Null[i],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Real[f], Char[g], < Real[h], Char[i], >*
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type ={ Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else: Null[c], Null[d],
Endif:
If: NAPL Boundary Type Option = { Dirichlet } { Zero Flux }
NAPL Pressure[e], Units[f] (Pa),
Elseif: NAPL Boundary Type Option = { Neumann }
NAPL Volumetric Flux[e], Units[f] (m/s),
Elseif: NAPL Boundary Type Option = { Hydraulic Gradient }
Base NAPL Pressure[e], Units[f] (Pa),
Else: Null[e], Null[f],
Endif:
If: Dissolved Oil Boundary Type Option = { Volumetric Concentration }
Dissolved Oil Volumetric Concentration[g], Units[h] (kg/m^3),
Elseif: Salt Boundary Type Option = { Aqueous Concentration }
Dissolved Oil Aqueous-Phase Volumetric Concentration[g], Units[h] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Volumetric }
Dissolved Oil Volumetric Concentration[g], Units[h] (kg/m^3),
Elseif: Salt Boundary Type Option = { Inflow Aqueous }
Dissolved Oil Aqueous-Phase Volumetric Concentration[g], Units[h] (kg/m^3),
Else: Null[g], Null[h],
Endif:
For: Number of Solutes:
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { NAPL Conc. }
Solute NAPL Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[i], Units[j] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow NAPL }
Solute NAPL Volumetric Concentration[i], Units[j] (1/m^3),
Else: Null[i], Null[j],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], <Real[i], Char[j],>*
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card =
{ Water-Oil-Dissolved Oil-Surfactant }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type ={ Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else:
Null[c], Null[d],
Endif:
If: NAPL Boundary Type Option = { Dirichlet } { Zero Flux }
NAPL Pressure[e], Units[f] (Pa),
Elseif: NAPL Boundary Type Option = { Neumann }
NAPL Volumetric Flux[e], Units[f] (m/s),
Elseif: NAPL Boundary Type Option = { Hydraulic Gradient }
Base NAPL Pressure[e], Units[f] (Pa),
Else:
Null[e], Null[f],
Endif:
If: Dissolved Oil Boundary Type Option = { Aqueous Concentration }
Dissolved Oil Aqueous-Phase Volumetric Concentration[g], Units[h] (kg/m^3),
Elseif: Disolved Oil Boundary Type Option = { Inflow Aqueous }
Dissolved Oil Aqueous-Phase Volumetric Concentration[g], Units[h] (kg/m^3),
Else:
Null[g], Null[h],
Endif:
If: Surfactant Boundary Type Option = { Aqueous Concentration }
Surfactant Aqueous-Phase Volumetric Concentration[i], Units[j] (kg/m^3),
Elseif: Surfactant Boundary Type Option = { Inflow Aqueous }
Surfactant Aqueous-Phase Volumetric Concentration[i], Units[j] (kg/m^3),
Else:
Null[i], Null[j],
Endif:

Boundary Conditions Card (contd)

For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { NAPL Conc. }
Solute NAPL Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[k], Units[l] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow NAPL }
Solute NAPL Volumetric Concentration[k], Units[l] (1/m^3),
Else:
Null[k], Null[l],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], Real[i], Char[j], <Real[k], Char[l],>*
Endfor: Number of Boundary Times

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[c], Units[d] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[c], Units[d] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[c], Units[d] (Pa),
Else:
Null[c], Null[d],
Endif:
Aqueous Dissolved Air Mass Fraction[e],
Aqueous Dissolved Oil Mass Fraction[f],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[g], Units[h] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[g], Units[h] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Base Gas Pressure[g], Units[h] (Pa),
Else:
Null[g], Null[h],
Endif:
Water Vapor Relative Humidity[i],
Oil Vapor Relative Humidity[j],
If: NAPL Boundary Type Option = { Dirichlet } { Zero Flux }
NAPL Pressure[k], Units[l] (Pa),
Elseif: NAPL Boundary Type Option = { Neumann }
NAPL Volumetric Flux[k], Units[l] (m/s),
Elseif: NAPL Boundary Type Option = { Hydraulic Gradient }
Base NAPL Pressure[k], Units[l] (Pa),
Else:
Null[k], Null[l],
Endif:

Boundary Conditions Card (contd)

For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Gas-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { NAPL Conc. }
Solute NAPL Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[m], Units[n] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow NAPL }
Solute NAPL Volumetric Concentration[m], Units[n] (1/m^3),
Else:
Null[m], Null[n],
Endif:
Endfor: Number of Solutes
Format: Real[a], Char[b], Real[c], Char[d], Real[e], Real[f], Real[g], Char[h],
Real[i], Real[j], Real[k], Char[l], < Real[m], Char[n], >
Endfor: Number of Boundary Times
Endif:

Boundary Conditions Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
For: Number of Boundary Times
Boundary Time[a], Units[b] (s),
If: Energy Boundary Type Option = { Dirichlet }
Temperature[c], Units[d] (C)
Elseif: Energy Boundary Type Option = { Neumann }
Energy Flux[c], Units[d] (W/m^2)
Else:
Null[c], Null[d],
Endif:
If: Aqueous-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Aqueous Pressure[e], Units[f] (Pa),
Elseif: Aqueous-Phase Boundary Type Option = { Neumann }
Aqueous Volumetric Flux[e], Units[f] (m/s),
Elseif: Aqueous-Phase Boundary Type Option = { Hydraulic Gradient }
Base Aqueous Pressure[e], Units[f] (Pa),
Else:
Null[e], Null[f],
Endif:
Aqueous Dissolved Air Mass Fraction[g],
Aqueous Dissolved Oil Mass Fraction[h],
If: Gas-Phase Boundary Type Option = { Dirichlet } { Zero Flux }
Gas Pressure[i], Units[j] (Pa),
Elseif: Gas-Phase Boundary Type Option = { Neumann }
Gas Volumetric Flux[i], Units[j] (m/s),
Elseif: Gas-Phase Boundary Type Option = { Hydraulic Gradient }
Base Gas Pressure[i], Units[j] (Pa),
Else:
Null[i], Null[j],
Endif:
Water Vapor Relative Humidity[k],
Oil Vapor Relative Humidity[l],
If: NAPL Boundary Type Option = { Dirichlet } { Zero Flux }
NAPL Pressure[m], Units[n] (Pa),
Elseif: NAPL Boundary Type Option = { Neumann }
NAPL Volumetric Flux[m], Units[n] (m/s),
Elseif: NAPL Boundary Type Option = { Hydraulic Gradient }
Base NAPL Pressure[m], Units[n] (Pa),
Else:
Null[m], Null[n],
Endif:

Boundary Conditions Card (contd)

For: Number of Solutes
If: Solute Transport Boundary Type Option = { Volumetric Conc. }
Solute Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Gas-Phase Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Aqueous Conc. }
Solute Aqueous-Phase Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { NAPL Conc. }
Solute NAPL Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Volumetric }
Solute Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Aqueous }
Solute Aqueous-Phase Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow Gas }
Solute Gas-Phase Volumetric Concentration[o], Units[p] (1/m^3),
Elseif: Solute Transport Boundary Type Option = { Inflow NAPL }
Solute NAPL Volumetric Concentration[o], Units[p] (1/m^3),
Else:
Null[o], Null[p],
Endif:
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Real[h], Real[i], Char[j], Real[k], Real[l], Real[m], Char[n], < Real[o], Char[p], >*
Endfor: Number of Boundary Times
Endif:

Endfor: Number of Boundary Condition Domains

Endcard: Boundary Conditions Card

A.20.1 Boundary Conditions Card Examples

```
#-----------------------------------------------------
~Boundary Conditions Card
#-----------------------------------------------------
2,
East,Hydraulic Gradient,Aqueous Concentration,
50,50,1,1,1,113,1,
0,day,183254,Pa,1.,1/ft^3,
West,Neumann,Outflow,
3,3,1,1,14,33,4,
0,min,-0.00021209,ft/sec,,,
37.4675,min,-0.00021209,ft/sec,,,
74.9232,min,-0.000200628,ft/sec,,,
112.379,min,-0.000201526,ft/sec,,,
```

```
~Boundary Conditions Card
2,
West,Dirichlet Energy,Dirichlet Aqueous,Dirichlet Gas,
1,1,1,1,1,1,1,
0,day,70,C,101330,Pa,0,101330,Pa,1,
East,Neumann Energy,Zero Flux Aqueous,Zero Flux Gas,
50,50,1,1,1,1,1,
0,Day,-100,W/m^2,,,,,,,
```

```
~Boundary Conditions Card
2,
West,Neumann,Aqueous Conc.,
1,1,1,1,1,10,1,
0,yr,6.6e-5,m/s,0.0,kg/m^3,
East,Hydraulic Gradient,Aqueous Conc.,
20,20,1,1,1,10,1,
0,yr,121557.98,Pa,36.5921,kg/m^3,
```

```
~Boundary Conditions Card
2,
Top,Neumann,Zero Flux,Inflow Aqueous,Inflow Aqueous,
1,1,1,1,92,92,1,
0,hr,-3.32,cm/hr,,,0.,kg/m^3,39.913,kg/m^3,39.913,kg/m^3,
Bottom,Dirichlet,Dirichlet,Outflow,Outflow,
1,1,1,1,1,1,1,
0,hr,111119.948,Pa,111119.948,Pa,,,,,
```

A.21 Source Card

Card Title[a] { ~Source [Card] }
Format: *Char*[a]

Number of Source Domains[a],
Format: Integer[a],

For: Number of Source Domains

If: Operational Mode Option: Solution Control Card = { Water }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ Well [No Storage] [No Volume] }
{ Solute [Density], Solute Name } { Solute IWVS, Solute Name }
If: Source Type Option = { IWVS }
Upper Screen Domain: I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g],
Lower Screen Surface Orientation Option[h],
{ Bottom } { Top } { West } { East } { South } { North }
Lower Screen Domain: I-Start Index[i], I-End Index[j], J-Start Index[k], J-End Index[l],
K-Start Index[m], K-End Index[n], Number of Source Times[o],
Format: *Char*[a], *Integer*[b], *Integer*[c], *Integer*[d], *Integer*[e], *Integer*[f], *Integer*[g],
Char[h], *Integer*[i], *Integer*[j], *Integer*[k], *Integer*[l], *Integer*[m], *Integer*[n], *Integer*[o],
Else:
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char*[a], *Integer*[b], *Integer*[c], *Integer*[d], *Integer*[e], *Integer*[f], *Integer*[g], *Integer*[h],
Endif:

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] } { Salt [Density] }
{ Well [No Storage] [No Volume] }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char*[a], *Integer*[b], *Integer*[c], *Integer*[d], *Integer*[e], *Integer*[f], *Integer*[g], *Integer*[h],

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
Source Type Option[a]
{ Aqueous Volumetric } { Aqueous Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Salt }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Salt [Density] }
{Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy }
Source Type Option[a]
{ Power [Density] } { Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy-Salt }
Source Type Option[a]
{ Power [Density] } { Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Salt [Density] } { Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ NAPL Volumetric [Density] } { NAPL Mass [Density] }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ NAPL Volumetric [Density] } { NAPL Mass [Density] }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card =
{ Water-Oil-Dissolved Oil-Surfactant }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ NAPL Volumetric [Density] } { NAPL Mass [Density] }
{ Surfactant Mass [Density] } { Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
Source Type Option[a]
{ Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ NAPL Volumetric [Density] } { NAPL Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
Source Type Option[a]
{ Power [Density] } { Aqueous Volumetric [Density] } { Aqueous Mass [Density] }
{ NAPL Volumetric [Density] } { NAPL Mass [Density] }
{ Gas Volumetric [Density] w/ Mass Fraction }
{ Gas Volumetric [Density] w/ Relative Humidity }
{ Gas Mass [Density] w/ Mass Fraction } { Gas Mass [Density] w/ Relative Humidity }
{ Solute [Density], Solute Name }
I-Start Index[b], I-End Index[c], J-Start Index[d], J-End Index[e],
K-Start Index[f], K-End Index[g], Number of Source Times[h],
Format: *Char[a], Integer[b], Integer[c], Integer[d], Integer[e], Integer[f], Integer[g], Integer[h],*
Endif:

If: Operational Mode Option: Solution Control Card = { Water }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Aqueous Volumetric }
Aqueous Volumetric Rate[c], Units[d] (m^3/s),
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Aqueous Mass }
Aqueous Mass Rate[c], Units[d] (kg/s),
Format: Real[a], Char[b], Real[c], Char[d],
Elseif: Source Type Option = { Aqueous Volumetric Density }
Aqueous Volumetric Density Rate[c], Units[d] (m^3/s m^3),
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Aqueous Mass Density }
Aqueous Mass Density Rate[c], Units[d] (kg/s m^3),
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Well [No Storage] [No Volume] }
Aqueous Mass Rate[c], Units[d] (kg/s),
Borehole Radius[e], Units[f] (m),
Well Bottom Pressure Constraint[g], Units[h] (Pa),
For: Number of Solutes
Solute Aqueous Concentration[i], Units[j] (1/m^3),
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], < Real[i], Char[j], >*
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s),
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s),
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Solute IWVS }
Exhaust Gas Temperature[c], Units[d] (C), Exhaust Gas Pressure[e], Units[f] (Pa),
Air/Water Volumetric Flow Ratio[g],
Air/Water Partition Coefficient (Henry's Constant)[h], Units[i] (Pa),
Vapor Stripping Efficiency[j],
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Real[h], Char[i], Real[j],*
Endif:
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Aqueous Volumetric }
Aqueous Volumetric Rate[c], Units[d] (m^3/s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Aqueous Mass }
Aqueous Mass Rate[c], Units[d] (kg/s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Aqueous Volumetric Density }
Aqueous Volumetric Density Rate[c], Units[d] (m^3/s m^3)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Aqueous Mass Density }
Aqueous Mass Density Rate[c], Units[d] (kg/s m^3)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Well [No Storage] [No Volume] }
Aqueous Mass Rate[c], Units[d] (kg/s),
Borehole Radius[e], Units[f] (m),
Well Bottom Pressure Constraint[g], Units[h] (Pa),
Salt Aqueous Concentration[i], Units[j] (kg/m^3),
For: Number of Solutes
Solute Aqueous Concentration[k], Units[l] (1/m^3),
Endfor: Number of Solutes
Format: *Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], Real[i], Char[j],*
< Real[k], Char[l], >
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Salt }
Salt Mass Rate[c], Units[d] (kg/s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Elseif: Source Type Option = { Salt Density }
Salt Mass Density Rate[c], Units[d] (kg/m^3 s)
Format: *Real[a], Char[b], Real[c], Char[d],*
Endif:
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Aqueous Volumetric }
Pressure[c], Units[d] (Pa), Aqueous Volumetric Rate[e], Units[f] (m^3/s),
Dissolved Air Mass Fraction[g],
Elseif: Source Type Option = { Aqueous Mass }
Pressure[c], Units[d] (Pa), Aqueous Mass Rate[e], Units[f] (kg/s),
Dissolved Air Mass Fraction[g],
Elseif: Source Type Option = { Aqueous Volumetric Density }
Pressure[c], Units[d] (Pa), Aqueous Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Dissolved Air Mass Fraction[g],
Elseif: Source Type Option = { Aqueous Mass Density }
Pressure[c], Units[d] (Pa), Aqueous Mass Density Rate[e], Units[f] (kg/s m^3),
Dissolved Air Mass Fraction[g],
Elseif: Source Type Option = { Gas Mass w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Mass Rate[e], Units[f] (kg/s),
Water Vapor Mass Fraction[g],
Elseif: Source Type Option = { Gas Mass w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Mass Rate[e], Units[f] (kg/s),
Water Vapor Relative Humidity[g],
Elseif: Source Type Option = { Gas Volumetric w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Volumetric Rate[e], Units[f] (m^3/s),
Water Vapor Mass Fraction[g],
Elseif: Source Type Option = { Gas Volumetric w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Volumetric Rate[e], Units[f] (m^3/s),
Water Vapor Relative Humidity[g],
Elseif: Source Type Option = { Gas Mass Density w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Mass Density Rate[e], Units[f] (kg/s m^3),
Water Vapor Mass Fraction[g],
Elseif: Source Type Option = { Gas Mass Density w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Mass Density Rate[e], Units[f] (kg/s m^3),
Water Vapor Relative Humidity[g],
Elseif: Source Type Option = { Gas Volumetric Density w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Water Vapor Mass Fraction[g],
Elseif: Source Type Option = { Gas Volumetric Density w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Water Vapor Relative Humidity[g],
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s),
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/s m^3),
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g],]*
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Power }
Power[c], Units[d] (W),
Elseif: Source Type Option = { Power Density }
Power Density[c], Units[d] (W/m^3),
Elseif: Source Type Option = { Aqueous Volumetric }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Aqueous Volumetric Rate[g], Units[h] (m^3/s), Dissolved Air Mass Fraction[i],
Elseif: Source Type Option = { Aqueous Mass }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Aqueous Mass Rate[g], Units[h] (kg/s), Dissolved Air Mass Fraction[i],
Elseif: Source Type Option = { Aqueous Volumetric Density }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Aqueous Volumetric Density Rate[g], Units[h] (m^3/s m^3), Dissolved Air Mass Fraction[i],
Elseif: Source Type Option = { Aqueous Mass Density }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Aqueous Mass Density Rate[g], Units[h] (kg/s m^3), Dissolved Air Mass Fraction[i],
Elseif: Source Type Option = { Gas Mass w/ Mass Fraction }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s), Water Vapor Mass Fraction[i],
Elseif: Source Type Option = { Gas Mass w/ Relative Humidity }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s), Water Vapor Relative Humidity[i]
Elseif: Source Type Option = { Gas Volumetric w/ Mass Fraction }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Volumetric Density Rate[g], Units[h] (m^3/s), Water Vapor Mass Fraction[i],
Elseif: Source Type Option = { Gas Volumetric w/ Relative Humidity }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Volumetric Rate[g], Units[h] (m^3/s), Water Vapor Relative Humidity[i],
Elseif: Source Type Option = { Gas Mass Density w/ Mass Fraction }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s m^3), Water Vapor Mass Fraction[i],
Elseif: Source Type Option = { Gas Mass Density w/ Relative Humidity }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s m^3), Water Vapor Relative Humidity[i]
Elseif: Source Type Option = { Gas Volumetric Density w/ Mass Fraction }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Volumetric Density Rate[g], Units[h] (m^3/s m^3), Water Vapor Mass Fraction[i],
Elseif: Source Type Option = { Gas Volumetric Density w/ Relative Humidity }
Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
Gas Volumetric Density Rate[g], Units[h] (m^3/s m^3), Water Vapor Relative Humidity[i],
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s),
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/s m^3),
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g], Char[h], Real[i],]*
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
  For: Number of Source Times
    Source Time[a], Units[b] (s),
    If: Source Type Option = { Aqueous Volumetric }
      Aqueous Volumetric Rate[c], Units[d] (m^3/s), Dissolved Oil Mass Fraction[e],
    Elseif: Source Type Option = { Aqueous Mass }
      Aqueous Mass Rate[c], Units[d] (kg/s), Dissolved Oil Mass Fraction[e],
    Elseif: Source Type Option = { Aqueous Volumetric Density }
      Aqueous Volumetric Density Rate[c], Units[d] (m^3/s m^3), Dissolved Oil Mass Fraction[e],
    Elseif: Source Type Option = { Aqueous Mass Density }
      Aqueous Mass Density Rate[c], Units[d] (kg/s m^3), Dissolved Oil Mass Fraction[e],
    Elseif: Source Type Option = { NAPL Volumetric }
      NAPL Volumetric Rate[c], Units[d] (m^3/s),
    Elseif: Source Type Option = { NAPL Mass }
      NAPL Mass Rate[c], Units[d] (kg/s),
    Elseif: Source Type Option = { NAPL Volumetric Density }
      NAPL Volumetric Density Rate[c], Units[d] (m^3/s m^3),
    Elseif: Source Type Option = { NAPL Mass Density }
      NAPL Mass Density Rate[c], Units[d] (kg/s m^3),
    Elseif: Source Type Option = { Solute }
      Solute Rate[c], Units[d] (1/s),
    Elseif: Source Type Option = { Solute Density }
      Solute Density Rate[c], Units[d] (1/s m^3),
    Endif:
    Format: *Real[a], Char[b], Real[c], Char[d], [Real[e],]*
  Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Aqueous Volumetric }
Aqueous Volumetric Rate[c], Units[d] (m^3/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Elseif: Source Type Option = { Aqueous Mass }
Aqueous Mass Rate[c], Units[d] (kg/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Elseif: Source Type Option = { Aqueous Volumetric Density }
Aqueous Volumetric Density Rate[c], Units[d] (m^3/s m^3),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Elseif: Source Type Option = { Aqueous Mass Density }
Aqueous Mass Density Rate[c], Units[d] (kg/s m^3),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Elseif: Source Type Option = { NAPL Volumetric }
NAPL Volumetric Rate[c], Units[d] (m^3/s),
Elseif: Source Type Option = { NAPL Mass }
NAPL Mass Rate[c], Units[d] (kg/s),
Elseif: Source Type Option = { NAPL Volumetric Density }
NAPL Volumetric Density Rate[c], Units[d] (m^3/s m^3),
Elseif: Source Type Option = { NAPL Mass Density }
NAPL Mass Density Rate[c], Units[d] (kg/s) ,
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s),
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s),
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g], Char[h],]*
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card =
{ Water-Oil-Dissolved Oil-Surfactant }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Aqueous Volumetric }
Aqueous Volumetric Rate[c], Units[d] (m^3/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Surfactant Concentration[g], Units[h] (kg/m^3),
Elseif: Source Type Option = { Aqueous Mass }
Aqueous Mass Rate[c], Units[d] (kg/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Surfactant Concentration[g], Units[h] (kg/m^3),
Elseif: Source Type Option = { Aqueous Volumetric Density }
Aqueous Volumetric Density Rate[c], Units[d] (m^3/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Surfactant Concentration[g], Units[h] (kg/m^3),
Elseif: Source Type Option = { Aqueous Mass Density }
Aqueous Mass Density Rate[c], Units[d] (kg/s),
Dissolved Oil Concentration[e], Units[f] (kg/m^3),
Surfactant Concentration[g], Units[h] (kg/m^3),
Elseif: Source Type Option = { NAPL Volumetric }
NAPL Volumetric Rate[c], Units[d] (m^3/s),
Elseif: Source Type Option = { NAPL Mass }
NAPL Mass Rate[c], Units[d] (kg/s),
Elseif: Source Type Option = { NAPL Volumetric Density }
NAPL Volumetric Density Rate[c], Units[d] (m^3/s m^3),
Elseif: Source Type Option = { NAPL Mass Density }
NAPL Mass Density Rate[c], Units[d] (kg/s m^3),
Elseif: Source Type Option = { Surfactant Mass }
Surfactant Mass Rate[c], Units[d] (kg/s),
Elseif: Source Type Option = { Surfactant Mass Density }
Surfactant Mass Density Rate[c], Units[d] (kg/m^3 s),
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s),
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s),
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g], Char[h],]*
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type= { Aqueous Volumetric }
Pressure[c], Units[d] (Pa), Aqueous Volumetric Rate[e], Units[f] (m^3/s),
Dissolved Air Mass Fraction[g], Dissolved Oil Mass Fraction[h],
Elseif: Source Type Option = { Aqueous Mass }
Pressure[c], Units[d] (Pa), Aqueous Mass Rate[e], Units[f] (kg/s),
Dissolved Air Mass Fraction[g], Dissolved Oil Mass Fraction[h],
Elseif: Source Type= { Aqueous Volumetric Density }
Pressure[c], Units[d] (Pa), Aqueous Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Dissolved Air Mass Fraction[g], Dissolved Oil Mass Fraction[h],
Elseif: Source Type Option = { Aqueous Mass Density }
Pressure[c], Units[d] (Pa), Aqueous Mass Density Rate[e], Units[f] (kg/s m^3),
Dissolved Air Mass Fraction[g], Dissolved Oil Mass Fraction[h],
Elseif: Source Type Option = { Gas Mass w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Mass Rate[e], Units[f] (kg/s),
Water Vapor Mass Fraction[g], Oil Vapor Mass Fraction[h],
Elseif: Source Type Option = { Gas Mass w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Mass Rate[e], Units[f] (kg/s),
Water Vapor Relative Humidity[g], Oil Vapor Relative Humidity[h],
Elseif: Source Type Option = { Gas Volumetric w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Volumetric Rate[e], Units[f] (m^3/s),
Water Vapor Mass Fraction[g], Oil Vapor Mass Fraction[h],
Elseif: Source Type Option = { Gas Volumetric w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Volumetric Rate[e], Units[f] (m^3/s),
Water Vapor Relative Humidity[g], Oil Vapor Relative Humidity[h],
Elseif: Source Type Option = { Gas Mass Density w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Mass Density Rate[e], Units[f] (kg/s m^3),
Water Vapor Mass Fraction[g], Oil Vapor Mass Fraction[h],
Elseif: Source Type Option = { Gas Mass Density w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Mass Density Rate[e], Units[f] (kg/s m^3),
Water Vapor Relative Humidity[g], Oil Vapor Relative Humidity[h],
Elseif: Source Type Option = { Gas Volumetric Density w/ Mass Fraction }
Pressure[c], Units[d] (Pa), Gas Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Water Vapor Mass Fraction[g], Oil Vapor Mass Fraction[h],
Elseif: Source Type Option = { Gas Volumetric Density w/ Relative Humidity }
Pressure[c], Units[d] (Pa), Gas Volumetric Rate[e], Units[f] (m^3/s m^3),
Water Vapor Relative Humidity[g], Oil Vapor Relative Humidity[h],

Source Card (contd)

Elseif: Source Type Option = { NAPL Volumetric }
Pressure[c], Units[d] (Pa), NAPL Volumetric Rate[e], Units[f] (m^3/s),
Elseif: Source Type Option = { NAPL Mass }
Pressure[c], Units[d] (Pa), NAPL Mass Rate[e], Units[f] (kg/s),
Elseif: Source Type Option = { NAPL Volumetric Density }
Pressure[c], Units[d] (Pa), NAPL Volumetric Density Rate[e], Units[f] (m^3/s m^3),
Elseif: Source Type Option = { NAPL Mass Density }
Pressure[c], Units[d] (Pa), NAPL Mass Density Rate[e], Units[f] (kg/s m^3),
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s)
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s)
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g], Real[h],]*
Endfor: Number of Source Times

Source Card (contd)

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
For: Number of Source Times
Source Time[a], Units[b] (s),
If: Source Type Option = { Power }
Power[c], Units[d] (W),
Elseif: Source Type Option = { Power Density }
Power Density[c], Units[d] (W/m^3),
Elseif: Source Type= { Aqueous Volumetric }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Aqueous Volumetric Rate[g], Units[h] (m^3/s),
Dissolved Air Mass Fraction[i], Dissolved Oil Mass Fraction[j],
Elseif: Source Type Option = { Aqueous Mass }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Aqueous Mass Rate[g], Units[h] (kg/s),
Dissolved Air Mass Fraction[i], Dissolved Oil Mass Fraction[j],
Elseif: Source Type= { Aqueous Volumetric Density }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Aqueous Volumetric Density Rate[g], Units[h] (m^3/s m^3),
Dissolved Air Mass Fraction[i], Dissolved Oil Mass Fraction[j],
Elseif: Source Type Option = { Aqueous Mass Density }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Aqueous Mass Density Rate[g], Units[h] (m^3/s m^3),
Dissolved Air Mass Fraction[i], Dissolved Oil Mass Fraction[j],
Elseif: Source Type Option = { Gas Mass w/ Mass Fraction }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Mass Rate[g], Units[h] (kg/s),
Water Vapor Mass Fraction[i], Oil Vapor Mass Fraction[j],
Elseif: Source Type Option = { Gas Mass w/ Relative Humidity }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Mass Rate[g], Units[h] (kg/s),
Water Vapor Relative Humidity[i], Oil Vapor Relative Humidity[j],
Elseif: Source Type Option = { Gas Volumetric w/ Mass Fraction }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Volumetric Rate[g], Units[h] (m^3/s),
Water Vapor Mass Fraction[i], Oil Vapor Mass Fraction[j],
Elseif: Source Type Option = { Gas Volumetric w/ Relative Humidity }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Volumetric Rate[g], Units[h] (m^3/s),
Water Vapor Relative Humidity[i], Oil Vapor Relative Humidity[j],
Elseif: Source Type Option = { Gas Mass Density w/ Mass Fraction }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s m^3),
Water Vapor Mass Fraction[i], Oil Vapor Mass Fraction[j],
Elseif: Source Type Option = { Gas Mass Density w/ Relative Humidity }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Mass Density Rate[g], Units[h] (kg/s m^3),
Water Vapor Relative Humidity[i], Oil Vapor Relative Humidity[j],

Source Card (contd)

Elseif: Source Type Option = { Gas Volumetric Density w/ Mass Fraction }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Volumetric Density Rate[g], Units[h] (m^3/s m^3),
Water Vapor Mass Fraction[i], Oil Vapor Mass Fraction[j],
Elseif: Source Type Option = { Gas Volumetric Density w/ Relative Humidity }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
Gas Volumetric Rate[g], Units[h] (m^3/s m^3),
Water Vapor Relative Humidity[i], Oil Vapor Relative Humidity[j],
Elseif: Source Type Option = { NAPL Volumetric }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa), NAPL Volumetric Rate[g], Units[h] (m^3/s),
Elseif: Source Type Option = { NAPL Mass }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa), NAPL Mass Rate[g], Units[h] (kg/s),
Elseif: Source Type Option = { NAPL Volumetric Density }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
NAPL Volumetric Density Rate[g], Units[h] (m^3/s m^3),
Elseif: Source Type Option = { NAPL Mass Density }
Temperature[c], Units[d], Pressure[e], Units[f] (Pa),
NAPL Mass Density Rate[g], Units[h] (kg/s m^3),
Elseif: Source Type Option = { Solute }
Solute Rate[c], Units[d] (1/s)
Elseif: Source Type Option = { Solute Density }
Solute Density Rate[c], Units[d] (1/m^3 s)
Endif:
Format: *Real[a], Char[b], Real[c], Char[d], [Real[e], Char[f], Real[g], Char[h], Real[i], Real[j],]*
Endfor: Number of Source Times

Endfor: Number of Source Domains

Endcard: Source Card

A.21.1 Source Card Examples

```
~Source Card
1,
Aqueous Volumetric,2,2,1,1,71,71,6,
0,min,0.319307,gal/min,
37.4675,min,0.319307,gal/min,
74.9232,min,0.30205,gal/min,
112.379,min,0.303402,gal/min,
149.835,min,0.307011,gal/min,
187.29,min,0.311148,gal/min,
```

```
~Source Card
1,
Well,1,1,1,1,1,20,1,
0,s,-0.5,ft^3/s,4.0,in,101325.0,Pa,
```

```
~Source Card
2,
Aqueous Volumetric,2,2,1,1,71,71,7,
0,min,0.8320722,gal/min,
66.046844,min,1.0441576,gal/min,
135.56322,min,1.0438042,gal/min,
205.09055,min,0.98627841,gal/min,
274.60406,min,0.99331003,gal/min,
344.13425,min,0.91488999,gal/min,
413.64774,min,1.0228366,gal/min,
Solute Iwvs,TCE,2,2,1,1,71,71,West,3,3,1,1,14,33,6,
0,min,20,C,92956,Pa,71.922127,47278556,Pa,0.95,
66.046844,min,20,C,92956,Pa,61.420078,47278556,Pa,0.95,
135.56322,min,20,C,92956,Pa,55.283066,47278556,Pa,0.95,
205.09055,min,20,C,92956,Pa,59.216434,47278556,Pa,0.95,
274.60406,min,20,C,92956,Pa,60.247459,47278556,Pa,0.95,
344.13425,min,20,C,92956,Pa,65.411583,47278556,Pa,0.95,
```

A.22 Output Control Card

Card Title[a] { ~Output [Control Card] }
Format: *Char*[a]

Number of Reference Nodes[a],
Format: *Integer*[a],

For: Number of Reference Nodes
I Index[a], J Index[b], K Index[c],
Endfor: Number of Reference Nodes
Format: *Integer*[a], *Integer*[b], *Integer*[c],

Reference Node Screen Output Frequency[a],
Reference Node Output File Frequency[b],
Output Time Units[c] (s),
Output Length Units[d] (m),
Screen Significant Digits[e],
Output File Significant Digits[f],
Plot File Significant Digits[g]
Format: *Integer*[a], *Integer*[b], *Char*[c], *Char*[d], *Integer*[e], *Integer*[f], *Integer*[g],

Number of Reference Node Variables[a],
Format: *Integer*[a],

For: Number of Reference Node Variables
Reference Node Variable Option[a], Reference Node Variable Units[b],
Format: *Real*[a], *Char*[b],
Endfor: Number of Reference Node Variables

Note: Refer to following pages for Reference Node Variable Options and Units.

Number of Plot File Times[a]
Format: *Integer*[a],

Output Control Card (contd)

For: Number of Plot File Times
Plot File Output Time[a], Units[b] (s)
Format: *Real*[a], *Char*[b],
Endfor: Number of Plot File Times

Number of Plot File Variables[a]
Format: *Integer*[a],

For: Number of Plot File Variables
Plot File Variable Option[a], Plot File Variable Units[b],
Format: *Char*[a], *Char*[b],
Endfor: Number of Plot File Variables

Note: Refer to the following pages for Plot File Variable Options and Units.

Endcard: Output Control Card

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

If: Operational Mode Option: Solution Control Card = { Water }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous fracture sat [uration] } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous matrix sat [uration] }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ aqueous well depth }* diffusive porosity } { effective trapped air sat [uration] }
{ gas fracture sat [uration] } { gas matrix-sat [uration] }
{ gas pressure } { gas sat [uration] }
{ phase condition } { rock/soil type }
{ solute conc. aqu [eous phase] } { solute conc. node [volume] }
{ solute mol. frac. aqu [eous phase] } { solute source [integral] }
{ temperature } { total water mass }
{ water aqueous conc [entration] } { water aqueous mass frac [tion]}
{ water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x solute flux [(lower surface)] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y solute flux [(lower surface)] }
{ ync aqueous vol [umetric flux (node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z solute flux [(lower surface)] }
{ znc aqueous vol [umetric flux (node centered)] }

* Reference Node Variable Only

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Salt }
{ air aqueous conc [entration] }{ air aqueous mass frac [tion] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous fracture sat [uration] } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous matrix sat [uration] }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ aqueous well depth }* { diffusive porosity }
{ effective trapped air sat [uration] } { gas fracture sat [uration] }
{ gas matrix-sat [uration] } { gas pressure }
{ gas sat [uration] } { phase condition } { rock/soil type }
{ salt aqueous conc [entration] }{ salt volumetric conc [entration] }
{ solute conc. aqu [eous phase] }{ solute conc. node [volume] }
{ solute mol. frac. aqu [eous phase] }{ solute source [integral] }
{ temperature }{ total water mass }
{ water aqueous conc [entration] }{ water aqueous mass frac [tion] }
{ water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x salt flux [(lower surface)] } { x solute flux [(lower surface)] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc salt flux [(node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y salt flux [(lower surface)] } { y solute flux [(lower surface)] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync salt flux [(node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z salt flux [(lower surface)] } { z solute flux [(lower surface)] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc salt flux [(node centered)] }

* Reference Node Variable Only

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas mass frac [tion] } { air gas mole frac [tion] }
{ air source [integral] } { apparent water sat [uration] }
{ aqueous density } { aqueous fracture sat [uration] }
{ aqueous gauge pressure } { aqueous hydraulic head }
{ aqueous matrix sat [uration] } { aqueous moisture cont [ent] }
{ aqueous pressure } { aqueous relative perm [eability] }
{ aqueous sat [uration] } { diffusive porosity }
{ effective trapped air sat [uration] } { gas air conc [entration] }
{ gas density } { gas fracture sat [uration] }
{ gas gauge pressure } { gas hydraulic head }
{ gas matrix sat [uration] } { gas pressure }
{ gas relative perm [eability] } { gas sat [uration] }
{ phase condition } { rock/soil type }
{ solute conc. aqu [eous] } { solute conc. gas }
{ solute conc. node } { solute mol. frac. aqu [eous] }
{ solute mol. frac. gas } { solute source [integral] }
{ temperature } { total air mass }
{ total water mass } { water aqueous conc [entration] }
{ water aqueous mass frac [tion] } { water gas conc [entration] }
{ water gas mass frac [tion] } { water gas mole frac [tion] }
{ water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x gas vol [umetric flux (lower surface)] }
{ x solute flux [(lower surface)] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y gas vol [umetric flux (lower surface)] }
{ y solute flux [(lower surface)] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z gas vol [umetric flux (lower surface)] }
{ z solute flux [(lower surface)] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Salt }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { air source [integral] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous fracture sat [uration] } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous matrix sat [uration] }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { effective trapped air sat [uration] }
{ gas density } { gas fracture sat [uration] }
{ gas gauge pressure } { gas hydraulic head }
{ gas matrix sat [uration] }
{ gas pressure } { gas relative perm [eability] }
{ gas sat [uration] } { phase condition } { rock/soil type }
{ salt aqueous conc [entration] } { salt volumetric conc [entration] }
{ solute conc. aqu [eous] } { solute conc. gas }
{ solute conc. node } { solute mol. frac. aqu [eous] }
{ solute mol. frac. gas } { solute source [integral] }
{ temperature } { total air mass } { total water mass }
{ water aqueous conc [entration] } { water aqueous mass frac [tion] }
{ water gas conc [entration] } { water gas mass frac [tion] }
{ water gas mole frac [tion] } { water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x gas vol [umetric flux (lower surface)] }
{ x salt flux [(lower surface)] }
{ x solute flux [(lower surface)] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] }
{ xnc salt flux [(node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y gas vol [umetric flux (lower surface)] }
{ y salt flux [(lower surface)] }
{ y solute flux [(lower surface)] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] }
{ ync salt flux [(node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z gas vol [umetric flux (lower surface)] }
{ z salt flux [(lower surface)] }
{ z solute flux [(lower surface)] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] }
{ znc salt flux [(node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { air source [integral] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous fracture sat [uration] } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous matrix sat [uration] }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { effective trapped air sat [uration] }
{ energy source [integral] } { gas density }
{ gas fracture sat [uration] } { gas gauge pressure }
{ gas hydraulic head } { gas matrix sat [uration] }
{ gas pressure } { gas relative perm [eability] }
{ gas sat [uration] } { ice density } { ice pressure }
{ ice sat [uration] } { phase condition } { rock/soil type }
{ solute conc. aqu [eous phase] } { solute conc. gas [phase] }
{ solute conc. node [volume] } { solute mol. frac. aqu [eous phase] }
{ solute mol. frac. gas [phase] } { solute source [integral] }
{ temperature } { total air mass } { total water mass }
{ water aqueous conc [entration] } { water aqueous mass frac [tion] }
{ water gas conc [entration] } { water gas mass frac [tion] }
{ water gas mole frac [tion] } { water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x gas vol [umetric flux (lower surface)] }
{ x heat flux [(lower surface)] } { x solute flux [(lower surface)] }
{ x thermal cond [uctivity] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] }
{ xnc heat flux [(node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y gas vol [umetric flux (lower surface)] }
{ y heat flux [(lower surface)] } { y solute flux [(lower surface)] }
{ y thermal cond [uctivity] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] }
{ ync heat flux [(node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z gas vol [umetric flux (lower surface)] }
{ z heat flux [(lower surface)] } { z solute flux [(lower surface)] }
{ z thermal cond [uctivity] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] }
{ znc heat flux [(node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Air-Energy-Salt }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { air source [integral] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous fracture sat [uration] } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous matrix sat [uration] }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { energy source [integral] }
{ gas density } { gas fracture sat [uration] }
{ gas gauge pressure } { gas hydraulic head }
{ gas matrix sat [uration] } { gas pressure }
{ gas relative perm [eability] } { gas sat [uration] }
{ ice density } { ice pressure } { ice sat [uration] }
{ phase condition } { rock/soil type }
{ salt aqueous conc [entration] } { salt volumetric conc [entration] }
{ solute conc. aqu [eous phase] } { solute conc. gas [phase] }
{ solute conc. node [volume] } { solute mol. frac. aqu [eous phase] }
{ solute mol. frac. gas [phase] } { solute source [integral] }
{ temperature } { total air mass } { effective trapped air sat [uration] }
{ total water mass } { water aqueous conc [entration] }
{ water aqueous mass frac [tion] } { water gas conc [entration] }
{ water gas mass frac [tion] } { water gas mole frac [tion] }
{ water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x gas vol [umetric flux (lower surface)] }
{ x heat flux [(lower surface)] } { x salt flux [(lower surface)] }
{ x solute flux [(lower surface)] } { x thermal cond [uctivity] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] }
{ xnc heat flux [(node centered)] } { xnc salt flux [(node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y gas vol [umetric flux (lower surface)] }
{ y heat flux [(lower surface)] } { y salt flux [(lower surface)] }
{ y solute flux [(lower surface)] } { y thermal cond [uctivity] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] }
{ ync heat flux [(node centered)] } { ync salt flux [(node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z gas vol [umetric flux (lower surface)] }
{ z heat flux [(lower surface)] } { z salt flux [(lower surface)] }
{ z solute flux [(lower surface)] } { z thermal cond [uctivity] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] }
{ znc heat flux [(node centered)] } { znc salt flux [(node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { apparent total sat [uration] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous gauge pressure } { aqueous hydraulic head }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { effective trapped air sat [uration] }
{ effective trapped oil sat [uration] } { napl density }
{ napl gauge pressure } { napl hydraulic head }
{ napl moisture cont [ent] } { napl pressure }
{ napl relative perm [eability] } { napl sat [uration] }
{ oil aqueous conc [entration] } { oil aqueous mass frac [tion] }
{ oil gas conc [entration] } { oil gas mass frac [tion] }
{ oil gas mole frac [tion] } { oil source [integral] }
{ phase condition } { rock/soil type }
{ solute conc. aqu [eous] } { solute conc. napl }
{ solute conc. node } { solute mol. frac. aqu [eous] }
{ solute mol. frac. napl } { solute source [integral] }
{ temperature } { total moisture cont [ent] }
{ total oil mass } { total sat [uration] }
{ total water mass } { water aqueous conc [entration] }
{ water aqueous mass frac [tion] } { water gas conc [entration] }
{ water gas mass frac [tion] } { water gas mole frac [tion] }
{ water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] }
{ x napl vol [umetric flux (lower surface)] }
{ x solute flux [(lower surface)] }
{ xnc aqueous vol [umetric flux (node centered)] }
{ xnc napl vol [umetric flux (node centered)] }
{ y aqueous vol [umetric flux (lower surface)] }
{ y napl vol [umetric flux (lower surface)] }
{ y solute flux [(lower surface)] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync napl vol [umetric flux (node centered)] }
{ z aqueous vol [umetric flux (lower surface)] }
{ z napl vol [umetric flux (lower surface)] }
{ z solute flux [(lower surface)] }
{ znc aqueous vol [umetric flux (node centered)] }
{ znc napl vol [umetric flux (node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { apparent total sat [uration] }
{ apparent water sat [uration] } { aqueous courant }
{ aqueous density } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous moisture cont }
{ aqueous pressure } { aqueous relative perm [eability] }
{ aqueous sat [uration] } { diffusive porosity }
{ dissolved-oil aqueous conc [entration] } { effective trapped air sat [uration] }
{ effective trapped oil sat [uration] } { napl courant [number] }
{ napl density } { napl gauge pressure }
{ napl hydraulic head } { napl moisture cont [ent] }
{ napl pressure } { napl relative perm [eability] }
{ napl sat [uration] } { oil aqueous conc [entration] }
{ oil aqueous mass frac [tion] } { oil gas conc [entration] }
{ oil gas mass frac [tion] } { oil gas mole frac [tion] }
{ oil source [mass] } { phase condition }
{ rock/soil type } { solute aqueous conc [entration] }
{ solute aqueous mole frac [tion] } { solute napl conc [entration] }
{ solute napl mole frac [tion] } { solute source }
{ solute volumetric conc [entration] } { temperature }
{ total moisture cont [ent] } { total oil mass }
{ total sat [uration] } { total water mass }
{ trapped oil sat [uration] } { water aqueous conc [entration] }
{ water aqueous mass frac [tion] } { water gas conc [entration] }
{ water gas mass frac [tion] } { water gas mole frac [tion] }
{ water source [mass] } { x aqueous vol [unmetric flux] }
{ x dissolved-oil flux } { x napl vol [unmetric flux] }
{ x solute flux } { xnc aqueous vol [unmetric flux] }
{ xnc dissolved-oil flux } { xnc napl vol [unmetric flux] }
{ y aqueous vol [unmetric flux] } { y dissolved-oil flux }
{ y napl vol [unmetric flux] } { y solute flux }
{ ync aqueous vol [unmetric flux] } { ync dissolved-oil flux }
{ ync napl vol [unmetric flux] } { z aqueous vol [unmetric flux] }
{ z dissolved-oil flux } { z napl vol [unmetric flux] }
{ z solute flux } { znc aqueous vol [unmetric flux] }
{ znc dissolved-oil flux } { znc napl vol [unmetric flux] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Dissolved Oil-Surfactant }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { apparent total sat [uration] }
{ apparent water sat [uration] } { aqueous courant }
{ aqueous density } { aqueous gauge pressure }
{ aqueous hydraulic head } { aqueous moisture cont }
{ aqueous pressure } { aqueous relative perm [eability] }
{ aqueous sat [uration] } { aqueous surfactant mass frac [tion] }
{ diffusive porosity } { dissolved-oil aqueous conc [entration] }
{ effective trapped air sat [uration] } { effective trapped oil sat [uration] }
{ napl courant [number] } { napl density }
{ napl gauge pressure } { napl hydraulic head }
{ napl moisture cont [ent] } { napl pressure }
{ napl relative perm [eability] } { napl sat [uration] }
{ oil aqueous conc [entration] } { oil aqueous mass frac [tion] }
{ oil gas conc [entration] } { oil gas mass frac [tion] }
{ oil gas mole frac [tion] } { oil source [mass] }
{ phase condition } { rock/soil type }
{ solute aqueous conc [entration] } { solute aqueous mole frac [tion] }
{ solute napl conc [entration] } { solute napl mole frac [tion] }
{ solute source } { solute volumetric conc [entration] }
{ surfactant aqueous conc [entration] } { surfactant volumetric conc [entration] }
{ temperature } { total moisture cont [ent] }
{ total oil mass } { total sat [uration] }
{ total water mass } { trapped oil sat [uration] }
{ water aqueous conc [entration] } { water aqueous mass frac [tion] }
{ water gas conc [entration] } { water gas mass frac [tion] }
{ water gas mole frac [tion] } { water source [mass] }
{ x aqueous vol [unmetric flux] } { x dissolved-oil flux }
{ x napl vol [unmetric flux] } { x solute flux }
{ x surfactant flux } { xnc aqueous vol [unmetric flux] }
{ xnc dissolved-oil flux } { xnc napl vol [unmetric flux] }
{ xnc surfactant flux } { y aqueous vol [unmetric flux] }
{ y dissolved-oil flux } { y napl vol [unmetric flux] }
{ y solute flux } { y surfactant flux }
{ ync aqueous vol [unmetric flux] } { ync dissolved-oil flux }
{ ync napl vol [unmetric flux] } { ync surfactant flux }
{ z aqueous vol [unmetric flux] } { z dissolved-oil flux }
{ z napl vol [unmetric flux] } { z solute flux }
{ z surfactant flux } { znc aqueous vol [unmetric flux] }
{ znc dissolved-oil flux } { znc napl vol [unmetric flux] }
{ znc surfactant flux }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { apparent total sat [uration] }
{ apparent water sat [uration] } { aqueous density }
{ aqueous gauge pressure } { aqueous hydraulic head }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { effective trapped air sat [uration] }
{ effective trapped oil sat [uration] } { gas density }
{ gas gauge pressure } { gas hydraulic head }
{ gas pressure } { gas relative perm [eability] }
{ gas sat [uration] } { napl density }
{ napl gauge pressure } { napl hydraulic head }
{ napl moisture cont [ent] } { napl pressure }
{ napl relative perm [eability] } { napl sat [uration] }
{ oil aqueous conc [entration] } { oil aqueous mass frac [tion] }
{ oil gas conc [entration] } { oil gas mass frac [tion] }
{ oil gas mole frac [tion] } { oil source [integral] }
{ phase condition } { rock/soil type }
{ solute conc. aqu [eous phase] } { solute conc. gas [phase] }
{ solute conc. napl [phase] } { solute conc. node [volume] }
{ solute mol. frac. aqu [eous phase] } { solute mol. frac. gas [phase] }
{ solute mol. frac. napl } { solute source [integral] }
{ temperature } { total air mass }
{ total moisture cont [ent] } { total oil mass }
{ total sat [uration] } { total water mass }
{ water aqueous conc [entration] } { water aqueous mass frac [tion] }
{ water gas conc [entration] } { water gas mass frac [tion] }
{ water gas mole frac [tion] } { water source [integral] }
{ x aqueous vol [umetric flux (lower surface)] } { x gas vol [umetric flux (lower surface)] }
{ x napl vol [umetric flux (lower surface)] }
{ x solute flux [(lower surface)] } { xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] } { xnc napl vol [umetric flux (node centered)] }
{ y aqueous vol [umetric flux (lower surface)] } { y gas vol [umetric flux (lower surface)] }
{ y napl vol [umetric flux (lower surface)] }
{ y solute flux [(lower surface)] } { ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] } { ync napl vol [umetric flux (node centered)] }
{ z aqueous vol [umetric flux (lower surface)] } { z gas vol [umetric flux (lower surface)] }
{ z napl vol [umetric flux (lower surface)] }
{ z solute flux [(lower surface)] } { znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] } { znc napl vol [umetric flux (node centered)] }

Output Control Card (contd)

Reference Node Variable and Plot File Variable Options

Elseif: Operational Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
{ air aqueous conc [entration] } { air aqueous mass frac [tion] }
{ air gas conc [entration] } { air gas mass frac [tion] }
{ air gas mole frac [tion] } { aqueous density }
{ aqueous gauge pressure } { aqueous hydraulic head }
{ aqueous moisture cont [ent] } { aqueous pressure }
{ aqueous relative perm [eability] } { aqueous sat [uration] }
{ diffusive porosity } { effective trapped air sat [uration] }
{ effective trapped oil sat [uration] } { energy source }
{ gas density } { gas gauge pressure }
{ gas hydraulic head } { gas pressure }
{ gas relative perm [eability] } { gas sat [uration] }
{ napl density } { napl gauge pressure }
{ napl hydraulic head } { napl moisture cont [ent] }
{ napl pressure } { napl relative perm [eability] }
{ napl sat [uration] } { oil aqueous conc [entration] }
{ oil aqueous mass frac [tion] } { oil gas conc [entration] }
{ oil gas mass frac [tion] } { oil gas mole frac [tion] }
{ oil source [integral] } { phase condition }
{ rock/soil type } { solute conc. aqu [eous phase] }
{ solute conc. gas [phase] } { solute conc. napl [phase] }
{ solute conc. node [volume] } { solute mol. frac. aqu [eous phase] }
{ solute mol. frac. gas [phase] } { solute mol. frac. napl }
{ solute source [integral] } { temperature }
{ total air mass } { total moisture cont [ent] }
{ total oil mass } { total sat [uration] }
{ total water mass } { water aqueous conc [entration] }
{ water aqueous mass frac [tion] } { water gas conc [entration] }
{ water gas mass frac [tion] } { water gas mole frac [tion] }
{ water source [integral] } { x aqueous vol [unmetric flux] }
{ x gas vol [unmetric flux] } { x heat flux }
{ x napl vol [unmetric flux] } { x solute flux }
{ x thermal cond [uctivity] } { xnc aqueous vol [umetric flux (node centered)] }
{ xnc gas vol [umetric flux (node centered)] } { xnc heat flux [(node centered)] }
{ xnc napl vol [umetric flux (node centered)] }
{ y aqueous vol [unmetric flux] } { y gas vol [unmetric flux] }
{ y heat flux } { y napl vol [unmetric flux] } { y solute flux } { y thermal cond [uctivity] }
{ ync aqueous vol [umetric flux (node centered)] }
{ ync gas vol [umetric flux (node centered)] } { ync heat flux [(node centered)] }
{ ync napl vol [umetric flux (node centered)] }
{ z aqueous vol [unmetric flux] } { z gas vol [unmetric flux]) }
{ z heat flux } { z napl vol [unmetric flux] } { z solute flux) } { z thermal cond [uctivity] }
{ znc aqueous vol [umetric flux (node centered)] } { znc gas vol [umetric flux (node centered)] }
{ znc heat flux [(node centered)] } { znc napl vol [umetric flux (node centered)] }
Endif:

Output Control Card (contd)

Default Units for the Reference Node and Plot File Variables

Air Concentration (kg/m^3)	Air Source Integral (kg)
Aqueous Relative Permeability (null)	Aqueous Sat [uration] (null)
Aqueous Volumetric Flux (m/s)	Density (kg/m^3)
Energy Source Integral (J)	Gas Relative Permeability (null)
Gas Sat [uration] (null)	Gas Volumetric Flux (m/s)
Gauge Pressure (Pa)	Head (m),
Heat Flux (W/m^2)	Mass (kg)
Mass Fraction (null)	Moisture Content (null)
Mole Fraction (null)	NAPL Relative Permeability (null)
NAPL Sat [uration] (null)	NAPL Volumetric Flux (m/s)
Phase Condition (null)	Pressure (Pa)
Relative Permeability (null)	Sat [uration] (null)
Salt Conc. (kg/m^3)	Salt Aqueous Conc. (kg/m^3)
Salt Flux (kg/m^2 s)	Solute Conc. (1/m^3)
Solute Mole Fraction (null)	Solute Flux (1/m^2 s)
Temperature (C)	Thermal Conductivity (W/m K)
Oil Concentration (kg/m^3)	Oil Source Integral (kg)
Water Concentration (kg/m^3)	Water Source Integral (kg)
Well Depth (m)	

A.22.1 Output Control Card Examples

```
~Output Options Card
3,
1,1,42,
1,1,62,
1,1,92,
10,10,hr,cm,6,6,6,
8,
aqueous sat [ uration ],,
apparent water sat [ uration ],,
NAPL sat [ uration ],,
oil aqueous concentration,mg/L,
surfactant aqueous concentration,mg/L,
trapped oil sat [ uration ],,
total trapping number,,
minimum effective aqueous sat [ uration ],,
3,
0.025,hr,
0.075,hr,
0.125,hr,
8,
aqueous sat [ uration ],,
apparent water sat [ uration ],,
NAPL sat [ uration ],,
oil aqueous concentration,mg/L,
surfactant aqueous concentration,mg/L,
trapped oil sat [ uration ],,
total trapping number,,
minimum effective aqueous sat [ uration ],,
```

```
~Output Control Card
4,
20,1,1,
20,1,3,
20,1,6,
20,1,10,
1,1,yr,m,,,,
4,
Salt Aqueous Concentration,kg/m^3,
Aqueous Pressure,Pa,
Aqueous Sat [ uration ],,
X Aqueous Volumetric Flux,m/s,
0,
4,
Salt Aqueous Concentration,kg/m^3,
Aqueous Density,kg/m^3,
Aqueous Pressure,Pa,
Aqueous Sat [ uration ],,
```

A.23 Surface Flux Card

Card Title[a] { ~Surface [Flux Card] }
Format: *Char*[a]

Number of Surface Flux Inputs[a]
Format: Integer[a],

For: Number of Surface Flux Inputs:

If: Operational Mode Option: Solution Control Card = { Water }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux } { Aqueous Mass Flux }
{ Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Salt }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux } { Aqueous Mass Flux } { Salt Flux }
{ Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Air }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux } { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Air-Salt }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux } { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { Salt Flux } { Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Air-Energy }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux} { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { Heat Flux } { Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Air-Energy-Salt }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux} { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { Heat Flux } { Solute Flux, Solute Name }

Surface Flux Card (contd)

Elseif: Operation Mode Option: Solution Control Card = { Water-Oil }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux} { Aqueous Mass Flux } { NAPL Volumetric Flux }
{ NAPL Mass Flux } { Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Oil-Air }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux} { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { NAPL Volumetric Flux } { NAPL Mass Flux }
{ Solute Flux, Solute Name }

Elseif: Operation Mode Option: Solution Control Card = { Water-Oil-Air-Energy }
Surface Flux Type Option[a],
{ Aqueous Volumetric Flux} { Aqueous Mass Flux } { Gas Volumetric Flux }
{ Gas Mass Flux } { NAPL Volumetric Flux } { NAPL Mass Flux }
{ Heat Flux } { Solute Flux, Solute Name }
Endif:

If: Surface Flux Type Option = { Heat Flux }
Units[b] (W), Units[c] (J),
ElseIf: Surface Flux Type Option = { Volumetric Flux }
Units[b] (m^3/s), Units[c] (m^3),
Elseif: Surface Flux Type Option = { Mass Flux }
Units[b] (kg/s), Units[c] (kg),
Elseif: Surface Flux Type Option = { Solute Flux }
Units[b] (sol/s), Units[c] (sol),
Endif:

Surface Flux Orientation Option[d]
{ West } { East }
{ South } { North }
{ Top } { Bottom }
I-Start Index[e], I-End Index[f],
J-Start Index[g], J-End Index[h],
K-Start Index[i], K-End Index[j],
Format: Char[a], Char[b], Char[c], Char[d], Integer[e], Integer[f], Integer[g], Integer[h], Integer[i], Integer[j],

Endfor: Number of Surface Flux Inputs

Endcard: Surface Flux Card

A.23.1 Surface Flux Card Examples

```
~Surface Flux Card
5,
Aqueous Volumetric,gal/min,gal,East,2,2,1,1,14,33,
Aqueous Volumetric,gal/min,gal,East,2,2,1,1,71,100,
Aqueous Volumetric,gal/min,gal,East,52,52,1,1,1,113,
Solute Flux,TCE,1/min,,East,2,2,1,1,14,33,
Solute Flux,TCE,1/min,,East,2,2,1,1,71,100,
```

```
~Surface Flux Card
1,
NAPL Volumetric Flux,cm^3/min,cm^3,Top,1,1,1,1,10,10,
```

```
~Surface Flux Card
4,
Solute Flux,Tc-99,1/yr,,Bottom,1,10,1,1,1,1,
Solute Flux,U-238,1/yr,,Bottom,1,10,1,1,1,1,
Solute Flux,Np-237,1/yr,,Bottom,1,10,1,1,1,1,
Solute Flux,Pu-239,1/yr,,Bottom,1,10,1,1,1,1,
```

Distribution

OFFSITE

D. Augustijn
Water Resources & Management Dept
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

M. Brusseau
429 Schantz Building
Soil Water and Environmental Sci. Dept
University of Arizona
Tucson, 85721 AZ

R. Cady
U. S. Nuclear Regulatory Commission
Office of Nuclear Regulatory Research
Mail Stop T9-B3
Washington, DC 20555

J. Dane
202 Funches Hall
Agronomy and Soils Department
Auburn University, AL 36849

M. Hassanizadeh
Faculty of Civil Engineering
Delft University of Technology
Stevinweg 1
2628 CN Delft
The Netherlands

W. Herkelrath
US Geological Survey
Water Resources Division
345 Middlefiled Road, MS 496
Melo Park, CA 94025

C. Hofstee
Universitaet Stuttgart
Institut fuer Wasserbau
Pfaffenwaldring 61
D-70550 Stuttgart (Vaihingen)
Germany

F. Leij
US Salinity Laboratory
450 West Big Springs Road
Riverside, CA 92507-4617

A. Leijnse
Department of Water Resources
Wageningen Agricultural University
NIeuwe Kanaal 11
6709 PA Wageningen
The Netherlands

T. J. Nicholson
U. S. Nuclear Regulatory Commission
Office of Nuclear Regulatory Research
Mail Stop T9-F33
Washington, DC 20555

M. Schroth
Department of Civil Engineering
Oregon State University
202 Apperson Hall
Corvallis, OR 97331-2302

R. van Genuchten
US Salinity Laboratory
450 West Big Springs Road
Riverside, CA 92507-4617

ONSITE

Pacific Northwest National Laboratory
(47)

N. J. Aimo
A. Chilakapati
C. R. Cole
M. J. Fayer
M. D. Freshley
G. W. Gee
T. J. Gilmore
C. T. Kincaid
P. D. Meyer
W. E. Nichols (5)
M. Oostrom (10)
M. L. Rockhold
C. S. Simmons
G. P. Streile
A. L. Ward
M. D. White (10)
M. D. Williams
S. K. Wurstner
Information Release (7)