PNNL-13108

Pacific Northwest
National Laboratory
Operated by Battelle for the
U.S. Department of Energy

Subsurface Transport Over Reactive Multiphases (STORM): A General, Coupled, Nonisothermal Multiphase Flow, Reactive Transport, and Porous Medium Alteration Simulator, Version 2

User's Guide

D. H. Bacon
M. D. White
B. P. McGrail

February 2000

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

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC06-76RLO 1830

Printed in the United States of America

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831;
prices available from (615) 576-8401.

Available to the public from the National Technical Information Service,
U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161

♻ This document was printed on recycled paper.
(9/97)

PNNL-13108

Subsurface Transport Over Reactive Multiphases (STORM): A General, Coupled, Nonisothermal Multiphase Flow, Reactive Transport, and Porous Medium Alteration Simulator, Version 2

User's Guide

D. H. Bacon
M. D. White
B. P. McGrail

February 2000

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

Pacific Northwest National Laboratory
Richland, Washington 99352

Summary

The U.S. Department of Energy must approve a performance assessment (PA) to support the design, construction, approval, and closure of disposal facilities for immobilized low-activity waste (ILAW) currently stored in underground tanks at Hanford, Washington. A critical component of the PA is to provide quantitative estimates of radionuclide release rates from the engineered portion of the disposal facilities. Computer simulations are essential for this purpose because impacts on groundwater resources must be projected to periods of 10,000 years and longer. The computer code selected for simulating the radionuclide release rates is the Subsurface Transport Over Reactive Multiphases (STORM) simulator.

The STORM simulator solves coupled conservation equations for component mass and energy that describe subsurface flow over aqueous and gas phases through variably saturated geologic media. The resulting flow fields are used to sequentially solve conservation equations for reactive aqueous phase transport through variably saturated geologic media. These conservation equations for component mass, energy, and solute mass are partial differential equations that mathematically describe flow and transport through porous media.

The STORM simulator solves the governing-conservation equations and constitutive functions using numerical techniques for nonlinear systems. The partial differential equations governing thermal and fluid flow processes are solved by the integral volume finite difference method. These governing equations are solved simultaneously using Newton-Raphson iteration. The partial differential equations governing reactive solute transport are solved using either an operator split technique where geochemical reactions and solute transport are solved separately, or a fully coupled technique where these equations are solved simultaneously.

The STORM simulator is written in the FORTRAN 77 language, following American National Standards Institute (ANSI) standards. The simulator utilizes a variable source code configuration that allows tailoring of the execution memory and speed to the specific problem by editing text parameter files and recompiling the code. Execution of the STORM simulator is controlled through a text input file. This input file uses a structured format of associated groups of input data. Input data files and results are also presented for model verification and example simulations. Appendix A to this report presents examples of the STORM Version 2 input data format for each of the groups.

Acronyms and Abbreviations

ANSI	American National Standards Institute
AREST-CT	TBS
DOE	Department of Energy
Ecology	Washington State Department of Ecology
Eh	electrode potential in volts
GMRES	Generalized Minimal Residual Method (accelerator)
IC	Incomplete Cholesky (preconditioner)
L	length
ILAW	immobilized low activity waste
LAW	low activity waste
M	mass
N/A	not applicable
NSPCG	nonsymmetric preconditioned conjugate gradient
ODE	ordinary differential equation
PA	performance assessment
PDE	partial differential equation
PNNL	Pacific Northwest National Laboratory
PUF	Pressurized Unsaturated Flow (experiments)
SI	Systeme Internationale
STDIO	standard input/output device
STOMP	Subsurface Transport Over Multiple Phases
STORM	Subsurface Transport Over Reactive Multiphases
STORM1	STORM mode with unsaturated flow
STORM2	STORM mode with two-phase flow
STORM3	STORM mode with two-phase flow and heat transfer
T	time
TVD	Total Variation Diminishing (transport scheme)
VOC	volatile organic carbon

Contents

Figures

Tables

Data Files

1.0 Introduction

The Hanford Site, in southeastern Washington State, has been used extensively to produce nuclear materials for the U.S. strategic defense arsenal by the Department of Energy (DOE) and its predecessors, the U.S. Atomic Energy Commission and the U.S. Energy Research and Development Administration. A large inventory of radioactive and mixed waste has accumulated in 177 buried single- and double shell tanks. Liquid waste recovered from the tanks will be pretreated to separate the low-activity fraction from the high-level and transuranic wastes (Ecology 1996). Vitrification is the leading option for immobilization of these wastes, expected to produce approximately 550,000 metric tons of Low Activity Waste (LAW) glass. This total tonnage, based on nominal Na_2O oxide loading of 20 percent by weight, is destined for disposal in a near-surface facility.

Before disposal of the immobilized waste can proceed, the DOE must approve a performance assessment, a document that describes the impacts, if any, of the disposal facility on public health and environmental resources. Results from a recent performance assessment are in a study published 2 years ago (Mann et al. 1998). A major conclusion from this study is that release rates of radionuclides from the glass waste form by reaction with water determines the impacts of the disposal action more than any other independent parameters. Consequently, a sound scientific basis for determining the long-term release rates of radionuclides from these glasses must be developed before this disposal action is accepted by regulatory agencies, stakeholders, and the public.

The rate of glass corrosion depends on temperature and the chemical composition of the water contacting it. Because glass dissolution itself changes the pH and solution composition, water percolating through a disposal system has neither a constant nor a uniform composition. Consequently, the glass corrosion rate in the disposal system also varies as a function of time and space. Over the last several years, DOE has supported the development of a numerical simulator at the Pacific Northwest National Laboratory (PNNL). This simulator can compute time and spatial variations in the chemical environment of unsaturated disposal systems in response to the corrosion of waste forms as well as other physical and chemical processes (McGrail and Bacon 1998). This report describes the latest accomplishments in the development of this computational tool, Subsurface Transport Over Reactive Multiphases (STORM), Version 2, a general, coupled non-isothermal multiphase flow and reactive transport simulator. It was developed by coupling STOMP, a non-isothermal multiphase flow simulator (White and Oostrom 1996), with AREST-CT Version 1.1, a reactive transport and porous medium alteration simulator (Chen et al. 1995; Chen et al. 1997). The general approach was coupling the two codes by making major modifications to AREST-CT subroutines, and modifying STOMP routines as little as possible.

The underlying mathematics in STORM is contained in a set of coupled, nonlinear, partial differential equations. They describe the rate of change of the solute concentrations of pore water in a variably saturated, non-isothermal porous medium, and the alteration of waste forms, packaging materials, backfill, and host rocks. The detailed mathematics is discussed elsewhere (Chen et al. 1996; White and Oostrom 1996). The following topics are included.

- kinetic dissolution of waste forms
- kinetic dissolution of host rocks
- kinetic precipitation and dissolution of secondary phases
- aqueous equilibrium speciation
- gas-aqueous equilibria
- redox reactions
- two-phase flow (water and air)
- dynamic hydraulic properties

Other physical and chemical features of the code are:

- 1-D (vertical Z-axis) or 2-D (X-Z Plane) solution domains
- general interface to take user specified chemistry and hydraulic properties
- non-isothermal multiphase flow and chemistry
- ionic strength correction according to modified B-dot equation
- spatially varying distribution of minerals in solid phase
- effective reaction surface varies with dissolution and precipitation of solids

The primary output of STORM, as a function of time and space, consists of the following:

- concentrations of aqueous species
- concentrations of gaseous species
- release flux of aqueous species and chemical components
- pH and Eh changes
- radii, surface areas, and volume fractions of solids
- dissolution/precipitation rates of solids
- production/consumption rates of aqueous species, including water
- temperature
- porosity and permeability changes
- aqueous phase saturations, velocities
- gas phase saturations, velocities

2.0 Governing Equations

2.1 Multiphase Flow and Heat Transfer

In broad terms, the STORM simulator solves coupled conservation equations for component mass and energy that describe subsurface flow over multiple phases through variably saturated geologic media. The resulting flow fields are used to sequentially solve conservation equations for reactive aqueous phase transport through variably saturated geologic media. These conservation equations for component mass, energy, and solute mass are partial differential equations. They mathematically describe flow and transport through porous media and are collectively entitled the governing equations. The STORM simulator has capabilities for modeling subsurface flow over two distinct phases: aqueous and gas. The aqueous phase primarily comprises liquid water with small quantities of dissolved air. The gas phase comprises variable amounts of air and water vapor.

It is possible to compile three modes of STORM: STORM1 (unsaturated flow), STORM2 (2-phase flow), and STORM3 (2-phase flow and heat transfer). STORM1 provides the fastest execution time if neither gas-phase flow or heat transfer is a major consideration.

2.1.1 Water Mass Conservation Equation

The water mass conservation equation, shown in Equation (2.1), equates the time rate of change of water mass within a control volume with the flux of water mass crossing the control volume surface. In the STORM simulator, water exists in the diffusive pore space as liquid water in the aqueous phase and as water vapor in the gas phase. Phase partitioning of water mass is computed assuming equilibrium conditions, implying the time scale for establishing thermodynamic equilibrium is significantly shorter than for component transport. Water transport occurs by advection through the aqueous and gas phases and also by diffusion-dispersion through the gas phase. Following the low solubility assumption for dissolved air in the aqueous phase, air diffusion-dispersion through the aqueous phase is neglected. Flow of fluid phases is computed from Darcy's law. Transport of phase components is computed from Fick's law.

$$\frac{\partial}{\partial t}\left[\sum_{\gamma=\ell,g}\left(\phi_D\,\omega_\gamma^w\,\rho_\gamma\,s_\gamma\right)\right]=-\sum_{\gamma=\ell,g}\left(\nabla\,\mathbf{F}_\gamma^w+\nabla\,\mathbf{J}_\gamma^w\right)+\dot{m}^w \tag{2.1}$$

where

$$\mathbf{F}_\gamma^w=-\frac{\omega_\gamma^w\,\rho_\gamma\,k_{r\gamma}\,\mathbf{k}}{\mu_\gamma}\left(\nabla\,P_\gamma+\rho_\gamma\,g\,\mathbf{z}_g\right)\;for\;\gamma=\ell,\,g \tag{2.2}$$

$$\mathbf{J}_\gamma^w = -\tau_\gamma \, \phi_D \, \rho_\gamma \, s_\gamma \frac{M^w}{M_\gamma} D_\gamma^w \, \nabla \, \chi_\gamma^w \; for\, \gamma = \ell, g \tag{2.3}$$

where ϕ_D = diffusive porosity

ω_γ^j = mole fraction of component j in phase γ

ρ_γ = phase density for phase γ, kg m^{-3}

s_γ = phase saturation for phase γ

$\mathbf{F}_\gamma^j$ = advective flux of component j in phase γ, kg m^{-2} s^{-1}

$\dot{m}^w$ = mass source rate of component j, kg m^{-3} s^{-1}

$\mathbf{J}_\gamma^j$ = diffusive-dispersive flux of component j for phase γ, kg m^{-2} s^{-1}

$\mathbf{k}$ = intrinsic permeability tensor, m^2

$k_{r\gamma}$ = fluid relative permeability of phase γ

P_γ = pressure of phase γ, Pa

$\mathbf{z}_g$ = unit gravitational direction vector

g = acceleration of gravity, m s^{-2}

μ_γ = kinematic viscosity of phase γ, Pa s

τ_γ = phase tortuosity for phase γ

M_γ = molecular weight of phase γ, kg mol^{-1}

D_γ^j = diffusion coefficient of component j for phase γ, m^2 s^{-1}

χ_γ^j = mole fraction of component j in phase γ

a = air component

w = water component

g = gas phase

ℓ = aqueous phase

t = time, s

2.1.2 Air Mass Conservation Equation

In a similar manner to the water mass conservation equation, the air mass conservation equation, shown in Equation (2.4), equates the time rate of change of the air mass within a control volume with the flux of air mass crossing the control volume surface. In the STORM simulator, air exists in the diffusive pore space as a component of the gas phase and is dissolved in the aqueous phase. Phase partitioning of air mass is computed assuming equilibrium conditions: this assumption implies the time scale for thermodynamic equilibrium in geologic media is significantly shorter than that for component transport. Air transport occurs by advection and diffusion dispersion through the aqueous and gas phases. Flow of fluid phases is computed from Darcy's law. Transport of phase components is computed from Fick's law.

$$\frac{\partial}{\partial t}\left[\sum_{\gamma=\ell,g}\left(\phi_D\,\omega_\gamma^a\,\rho_\gamma\,s_\gamma\right)\right]=-\sum_{\gamma=\ell,g}\left(\nabla\,\mathbf{F}_\gamma^a+\nabla\,\mathbf{J}_\gamma^a\right)+\dot{m}^a \tag{2.4}$$

where

$$\mathbf{F}_\gamma^a=-\frac{\omega_\gamma^a\,\rho_\gamma\,k_{r\gamma}\,\mathbf{k}}{\mu_\gamma}\left(\nabla\,P_\gamma+\rho_\gamma\,g\,\mathbf{z}_g\right)\,for\,\gamma=\ell,\,g \tag{2.5}$$

$$\mathbf{J}_\gamma^a=-\tau_\gamma\,\phi_D\,\rho_\gamma\,s_\gamma\,\frac{M^a}{M_\gamma}\,D_\gamma^a\,\nabla\,\chi_\gamma^a\;for\,\gamma=\ell,\,g \tag{2.6}$$

2.1.3 Energy Conservation Equation

The energy conservation equation, shown in Equation (2.7), equates the time rate of change of energy within a control volume with the flux of energy crossing the control volume surface. In the STORM simulator, thermal energy is partitioned, according to thermal equilibrium conditions, among the fluid and solid phases. The thermal capacitance of unconnected pore space, represented by the difference between the total and diffusive porosity, is computed as it is filled with liquid water. Heat transfer occurs by advection of phase mass, diffusion of component mass, and thermal diffusion through the fluid and solid phases. Heat transfer by hydraulic dispersion of flowing fluid phases is neglected. Enhanced vapor transport is incorporated through enhancement factors for component diffusion through the gas phase. Energy associated with component mass sources is included as internal heat generation sources. Reference states for enthalpy and internal energy are component dependent. Latent heat transport is considered through vapor transport using the gas phase and equilibrium thermodynamics.

$$\begin{aligned}&\frac{\partial}{\partial t}\left[\sum_{\gamma=\ell,g}\left(\phi_D\,\rho_\gamma\,s_\gamma\,u_\gamma\right)+\left(1-\phi_T\right)\rho_s\,u_s+\left(\phi_T-\phi_D\right)\rho_\ell\,u_\ell\right]=\\&-\sum_{\gamma=\ell,g}\left(\nabla\left(\rho_\gamma\,h_\gamma\,\mathbf{q}_\gamma\right)+\sum_{j=w,a}\left(\nabla\,h^j\,\mathbf{J}_\gamma^j\right)\right)+\nabla\left(\mathbf{k}_e\,\nabla T\right)+\sum_{j=w,a}\left(h^j\,\dot{m}^j\right)+\dot{Q}\end{aligned} \tag{2.7}$$

where the Darcy velocity is given by

$$\mathbf{q}_\gamma=-\frac{k_{r\gamma}\,\mathbf{k}}{\mu_\gamma}\left(\nabla\,P_\gamma+\rho_\gamma\,g\,\mathbf{z}_g\right)\,for\,\gamma=\ell,\,g \tag{2.8}$$

and the diffusive flux by

$$\mathbf{J}_{\gamma}^{j} = -\tau_{\gamma}\,\phi_{D}\,\rho_{\gamma}\,s_{\gamma}\,\frac{M^{j}}{M_{\gamma}}\,D_{\gamma}^{j}\,\nabla\,\chi_{\gamma}^{j}\ \textit{for}\ \gamma = \ell, g\ \textit{and}\ j = w, a \tag{2.9}$$

where ϕ_T = total porosity
h_j = enthalpy of component j, J kg^{-1}
h_γ = enthalpy of phase γ, J kg^{-1}
$\mathbf{k}_e$ = equivalent thermal conductivity tensor, W m^{-1} K^{-1}
T = temperature, K
$\dot{Q}$ = thermal energy source, W
μ_γ = kinematic viscosity of phase γ, Pa s
μ_γ = internal energy of phase γ, J kg^{-1}

2.2 Solute Transport

2.2.1 Solute Mass Conservation Equation

The solute mass conservation equation, shown in Equation (2.10), equates the time rate for change of solute mass within a control volume with the flux of solute mass crossing the control volume surface. The solute mass conservation equation is solved when an *Operator Split* solution scheme is used. In the STORM simulator, solute mass is partitioned among the fluid and solid phases assuming thermodynamic and geochemical equilibrium conditions. This assumption implies the time scales for thermodynamic and geochemical equilibrium are short compared against those for solute transport. For geologic media, this assumption is generally appropriate. Solute transport occurs by advection and diffusion-dispersion through the aqueous phase and gas phase. Radioactive chain decay of solutes is computed using first-order decay kinetics. Solute mass conservation equations are solved sequentially with progeny solutes preceding parent solutes. Progeny solute production rates are computed from parent decay rates and chain decay fractions with

$$\frac{\partial C}{\partial t} = -\sum_{\gamma=l,g}\left(\nabla\left[C_{\gamma}q_{\gamma}\right]\right) + \dot{m}^{C} - \dot{R}^{C}C + \sum_{\gamma=l,g}\left(\nabla\left[\left(\tau_{\gamma}\,s_{\gamma}\phi_{D}D_{\gamma}^{C} + s_{\gamma}\phi_{D}\mathbf{D}_{h\gamma}\right)\nabla C_{\gamma}\right]\right) \tag{2.10}$$

where C = mass of solute per unit volume of porous media, m^{-3}
C_γ = mass of solute per unit volume of either liquid or gas, m^{-3}
D_γ^C = solute diffusion coefficient for phase γ, m^2 s^{-1}
$\mathbf{D}_{h\gamma}$ = hydraulic dispersion tensor for phase γ, m^2 s^{-1}
$\dot{m}_C$ = solute source rate, s^{-1}
$\dot{R}^C$ = solute decay rate constant, s^{-1}

2.2.2 Aqueous Species Mass Conservation Equation

The aqueous species mass conservation equation is written as follows

$$\frac{\partial \phi_D \rho_\ell s_\ell C_i^\ell}{\partial t} = -\nabla \cdot F_i^\ell + S_i^r \tag{2.11}$$

where C_i^ℓ is the concentration of species i in the aqueous phase (denoted by ℓ) in mol kg^{-1}, F_i^ℓ is the solute flux of species i in the aqueous phase, S_i^r is the source/sink term of species i due to chemical reactions in mol m^{-3} s^{-1}. The aqueous species mass conservation equation is solved when a *Fully Coupled* solution scheme is used.

If it is assumed that the phenomenon of hydrodynamic dispersion can be represented by a Fickian type law, then the total solute flux due to advection, dispersion, and diffusion is given by

$$\mathbf{F}_i^S = \phi_D \rho_\ell s_\ell \left(C_i \mathbf{V}_\ell - \mathbf{D}(s_\ell) \cdot \nabla C_i \right) \tag{2.12}$$

where **D** is the hydrodynamic dispersion tensor of species *i* in the aqueous phase, m^2 s^{-1}.

The flow fields resulting from the air, water, and energy conservation equations are coupled with the solute mass conservation equations via transient changes in water saturation and average linear pore water velocity. This velocity is related to the Darcy velocity by

$$\mathbf{V}_\ell = \frac{\mathbf{q}_\ell}{\phi_D s_\ell} \ . \tag{2.13}$$

2.2.3 Coupled Aqueous and Gaseous Species Mass Conservation Equation

The mass conservation law for a particular species can be written as the sum of transport in the gas and aqueous phases:

$$\frac{\partial}{\partial t}\rho_\ell \phi_D \left(s_g C_i^g + s_l C_i^\ell\right) = -\nabla \bullet F_i^S + S_i^r \tag{2.14}$$

where F_i^S is the total flux of species i in both the gas and aqueous phases.

The total flux due to advection, dispersion, and diffusion in both the aqueous and gas phases is given by

$$F_i^S = \phi_D s_g \left(\mathbf{V}_g C_i^g - \mathbf{D}_i^g \cdot \nabla C_i^g\right) + \phi_D s_\ell \left(\mathbf{V}_\ell C_i^\ell - \mathbf{D}_i^\ell \cdot \nabla C_i^\ell\right) \tag{2.15}$$

where V_g is the pore-gas velocity and $\mathbf{D}_i^g$ is the hydrodynamic dispersion tensor for species i in the gas phase. The coupled aqueous and gaseous species mass conservation equation is solved when a *Fully Coupled* solution scheme is used.

Assuming equilibrium between the gaseous phase and aqueous phase, the concentration of the dissolved gas component in the aqueous phase can be expressed as a linear function of the concentration (or mole fraction or partial pressure) of that component in the gaseous phase through Henry's constant

$$C_i^\ell = K_H p_i^g = K_H^{'} C_i^g \tag{2.16}$$

where C_i^ℓ = aqueous phase concentration of the dissolved gas component

p_g = partial pressure of the component in the gas phase

C_i^g = gas phase concentration of the component

K_H = Henry's law constants expressed as a function of partial pressure

$K_H^{'}$ = Henry's law constants expressed as a function of mole fraction

Substituting (2.15) and (2.16) into (2.14) gives the mass conservation law in terms of the aqueous phase concentration of each species.

$$\begin{gathered}\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K_H^{'-1} + s_\ell\right) C_i^\ell\right] = S_i^r + \\ \nabla \bullet \left[-\phi_D \rho_\ell \left(s_g \mathbf{V}_g K_H^{'-1} + s_\ell \mathbf{V}_\ell\right) C_i^\ell + \phi_D \rho_\ell \left(s_g \mathbf{D}_i^g K_H^{'-1} + s_\ell \mathbf{D}_i^\ell\right) \cdot \nabla C_i^\ell\right]\end{gathered} \tag{2.17}$$

A series of transformations are applied to Equation (2.17) to eliminate terms associated with equilibrium chemical reactions. Each aqueous species is defined as a reactive species, a non-reactive species, or an equilibrium species. Because of the transformations applied, equilibrium species must be transported at the same rate as the linearly related reactive species. A different Henry's Law constant and dispersion coefficient is allowed only for each reactive aqueous species that is in equilibrium with a gas species.

2.3 Geochemical Reactions

A chemical reaction generally can be written as

$$v_{mj} B_m + \sum_{i=1}^{N_s} v_{ij} A_i = 0 \tag{2.18}$$

where B_m = solid m
A_i = aqueous species i
v_{ij} = stoichiometric coefficient of aqueous species i in reaction j (negative for reactants, positive for products)
v_{mj} = stoichiometric coefficient of solid m in reaction j (negative for dissolution, positive for precipitation, zero for pure aqueous reaction)

If the rate of the reaction in unit bulk volume is shown as W_j (in general a nonlinear function of the aqueous species concentrations), the rate of aqueous species i in that reaction is $v_{ij} W_j$. (This nonlinear function is defined in Section 2.3.2.) The source/sink term due to reactions in Equation (2.14) is then

$$S_i^r = \sum_{j=1}^{N_r} v_{ij} W_j \tag{2.19}$$

where N_r is the number of reactions.

2.3.1 Equilibrium and Kinetic Reactions

Geochemical models usually assume some reactions to be in equilibrium. This assumption is often justified for some reactions, especially those involving only aqueous species. However, solid dissolution/precipitation reactions can be quite slow and should be treated as kinetic. Furthermore, the equilibrium approach cannot capture some metastable solid phases, such as amorphous solids, that in many cases are intermediate products of reaction. For this reason, STORM has been designed to treat both equilibrium and kinetic reactions separately. To this end, the source/sink term is split, due to chemical reactions, into equilibrium and kinetic parts.

$$S_i^r = \sum_{j=1}^{N_e} v_{ij} W_j^e + \sum_{j=1}^{N_k} v_{ij} W_j^k \tag{2.20}$$

where W_j^e = the rate of equilibrium reaction j (mol m^{-3} s^{-1})

W_j^k = the rate of kinetic reaction j (mol m^{-3} s^{-1})

N_e = the number of equilibrium reactions

N_k = the number of kinetic reactions

$N_r = N_e + N_k$.

Equilibrium reactions are by no means zero-rate reactions. In fact, equilibrium reactions have high reaction rates and reach equilibrium quickly when transport, other reactions, or changes in physical-chemical conditions disturb it. Specifically, if the rate of a reaction is much greater than the characteristic time of the problem being solved, it should be classified as equilibrium; otherwise it should be classified as kinetic.

2.3.2 Rate Law of Kinetic Reactions

For any kinetic reaction, assuming its rate law is of the mass-action type, two equivalent ways can define the reaction rate: one is to use the forward rate and backward rate; the other is to use the forward rate and the equilibrium constant. In STORM, the latter approach is used. Thus,

$$W_j = A_{m(j)} k_j \left[\prod_{v_{ij}<0} (\gamma_i c_i)^{\frac{|v_{ij}|}{f_j}} - \prod_{v_{ij}>0} (\gamma_i c_i)^{\frac{v_{ij}}{f_j}} \Big/ \left(K_j^{eq}\right)^{1/f_j} \right] \tag{2.21}$$

where $A_{m(j)}$ = a factor that for aqueous reactions equals 1, and for surface-limited reactions it equals the effective reaction area in unit bulk volume

k_j = the rate constant of reaction j

K_j^{eq} = the equilibrium constant of reaction j

γ_i = the activity coefficient of species i

f_j = the factor of the reaction order

W_j = generally a nonlinear function of concentrations of aqueous species

Several formulas, such as the Davies equation (Davies 1962), the B-dot equation (Helgeson 1969), and Pitzer's model (Pitzer 1973), can be used to calculate activity coefficients. The B-dot equation, with modifications for neutral species (Wolery 1992), is computationally economic and stable for problems of moderately high salinity. Although Pitzer's model can handle high ionic strength, the lack of constants for all but a few aqueous species limits its applicability. Thus, the B-dot equation is used to calculate activity coefficients in STORM.

2.3.3 Solid Reactions

In the subsurface, chemical reactions change not only the concentration of solutes but also the properties of the porous media. Changes in the volume fraction and surface area of solids, porosity and permeability result in changes in the fluid flow field, solute transport, and chemical reactions. Interactions may take place through several paths and in some cases be very strong. Ignoring the interactions will not only decrease the accuracy of the simulations but in many cases, also will yield even qualitatively incorrect predictions. To fully account for the interactions between the fluid phases and the porous media, it is necessary to consider dynamic changes in the properties of porous media as a result of mass redistribution from chemical reactions. The first step is to introduce variables that describe porous media texture.

STORM uses two primary variables to describe a mineral: the volume fraction V_m and one or more sizes of variables (depending on the shape of the mineral grains). The volume fractions of minerals and the porosity of the porous medium obey the relation

$$\sum_{m=1}^{N_m} V_m + \phi_T = 1 \tag{2.22}$$

From the volume fraction and size variables, the number density n_m (number of grains of mineral m in unit bulk volume of porous media) is derived and the specific surface area A_m (surface area of mineral m in unit bulk volume of porous media) is determined when the shape of the grains is specified. For spherical grains, the size variable is the radius R_m, so that

$$n_m = \frac{3V_m}{4\pi R_m^3} \tag{2.23}$$

and

$$A_m = 4\pi R_m^2 n_m \tag{2.24}$$

By assuming that a dissolution/precipitation reaction is a surface-limited kinetic reaction, observing the rate law of Equation (2.21), the change rate of the volume fraction is

$$\frac{\partial V_m}{\partial t} = 4\pi R_m^2 n_m \frac{\partial R_m}{\partial t} \tag{2.25}$$

For general cases of more than one dissolution/precipitation reaction occurring for one mineral, the rate of change of the mineral radius is

$$\frac{\partial R_m}{\partial t} = \sum_{j=1}^{N_k} \frac{k_j}{\rho_m} \left[\prod_{v_{ij}<0} (\gamma_i c_i)^{\frac{|v_{ij}|}{f_j}} - \prod_{v_{ij}>0} (\gamma_i c_i)^{\frac{v_{ij}}{f_j}} \Big/ \left(K_j^{eq}\right)^{1/f_j} \right] \tag{2.26}$$

where ρ_m is the molar density of mineral m (mol m^{-3}), and the summation is over all kinetic reactions pertaining to mineral m.

Substituting Equation (2.24) into Equation (2.21) gives

$$W_j^k = 4\pi R_{m(j)}^2 n_{m(j)} k_j \left[\prod_{v_{ij}<0} (\gamma_i c_i)^{\frac{|v_{ij}|}{f_j}} - \prod_{v_{ij}>0} (\gamma_i c_i)^{\frac{v_{ij}}{f_j}} \Big/ \left(K_j^{eq}\right)^{1/f_j} \right] \tag{2.27}$$

that serves as a bridge between the aqueous phase and solid phases so that the whole system is mass-conservative.

2.4 Coupling Between Fluid Flow and Geochemical Reactions

2.4.1 Porosity Changes Coupled with Multiphase Flow

Porosity changes due to mineral dissolution and precipitation could have a significant effect on the flow regime surrounding the disposal facility. Therefore, porosity changes, calculated by the STORM texture solver, are coupled with the multiphase flow solver. Dissolution and precipitation reactions can change the volume fractions of solids as a function of time, thus changing the porosity as described in

Section 2.3.3. Furthermore, intrinsic permeability varies as a function of porosity. Two options for modeling this relationship are available in STORM. One option is the Fair and Hatch (1933) formula, developed from dimensional considerations and verified experimentally,

$$k = \frac{1}{\beta}\left[\frac{(1-\phi_T)^2}{\phi_T^3}\left(\frac{\alpha}{100}\sum_{(m)}\frac{P_m}{d_m}\right)^2\right]^{-1} \tag{2.28}$$

where k = intrinsic permeability in cm^2
β = packing factor, found experimentally to be about 5
α = sand shape factor, varying from 6.0 for spherical grains to 7.7 for angular ones
P_m = percentage of sand held between adjacent sieves
d_m = geometric mean diameter of adjacent sieves

The second option is the Kozeny-Carmen equation that was obtained from a theoretical derivation of Darcy's law and includes numerical coefficients that must be determined empirically,

$$k = C_0 \frac{\phi_T^3}{(1-\phi_T)^2 M_S^2}, \tag{2.29}$$

where M_S = specific surface area of the porous matrix (defined per unit volume of solid)
C_o = coefficient for which Carman (1937) suggested the value of 1/5

This modification was tested on a 3-node problem to ensure that water mass was being conserved as porosity varied during glass dissolution.

2.4.2 Water Consumption in Chemical Reactions Coupled with Multiphase Flow

Consumption of water varied during dissolution of waste glass, such as the following hydrolysis reaction for LD6-5412 glass:

$$\begin{aligned}
&Glass + 0.5OH^- + 0.45H_2O = 0.0202K^+ + 0.4208Na^+ 0.046Ca^{2+} + \\
&0.154Al(OH)_{3(aq)} + 0.094BO_2^- + 0.2712HSiO_3^- + 0.3348H_2SiO_4^{2-} + \\
&2.2795x10^{-8}SeO_3^{2-} + 1.6172x10^{-6}TcO_4^{2-} + 3.8318x10^{-8}IO_3^- + \\
&2.7213x10^{-5}U(OH)_{4(aq)} + 1.9265x10^{-9}Pu(OH)_5^- + 2.7251x10^{-9}NpO_2(OH)_{(aq)}
\end{aligned} \tag{2.30}$$

The consumption/production of water during precipitation of hydrated minerals may be large enough to affect unsaturated flow in the region of the waste repository. The STORM code couples unsaturated flow with water production/consumption by chemical reactions through the mass source rate of water in Equation (2.1).

$$\dot{m}^w = \sum_{j=1}^{N_r} \nu_{wj} W_j / \Omega \tag{2.31}$$

where N_r is the total number of reactions, ν_{wj} is the stoichiometric coefficient of water in reaction j, W_j is the rate of reaction per unit bulk volume, given by Equation (2.21), and Ω is the number of moles of solvent water comprising a mass of 1 kg (Ω=55.51). The water source/sink term due to reactions is added to any other water source/sinks specified in the STORM input file.

2.4.3 Gas Consumption in Chemical Reactions Coupled with Multiphase Flow

Consumption of gaseous species, particularly oxygen, may occur at large enough rates to affect the flow of air in the vadose zone. The STORM code couples unsaturated flow with gas production/consumption by chemical reactions through the mass source rate of air in Equation (2.4).

$$\dot{m}^a = \sum_{j=1}^{N_r} \nu_{gj} W_j M_j \tag{2.32}$$

where N_r is the total number of reactions, ν_{gj} is the stoichiometric coefficient of the gaseous species in reaction j, W_j is the rate of reaction per unit bulk volume, and M_j is the molecular weight of the gaseous species. The gas source/sink term due to reactions is added to any other gas source/sinks specified in the STORM input file.

3.0 Numerical Solution Methods

3.1 Introduction

The STORM simulator solves the governing-conservation equations and constitutive functions using numerical techniques for nonlinear systems. This section documents the transformation of the governing-conservation equations from partial-differential form to algebraic form, algebraic expression of boundary conditions, linearization of the conservation equations and constitutive functions, and solution of linear systems.

The governing-conservation equations for water, air, and energy conservation are discretized to algebraic form following the integrated-finite-difference method of (Patankar 1980). This transformation requires the physical domain to be spatially discretized into an orthogonal computational domain comprised of non-overlapping volumes (nodes). Each volume can contain a maximum of two neighboring nodes for each dimensional direction. Intrinsic properties are assumed to be uniform over the volume domain and are defined for a node point at the geometric center of the volume. Flux quantities are defined at the geometric center of the surfaces between node volumes and along a direction parallel to the surface normal. Fluxes across node surfaces between neighboring inactive nodes or adjacent to the domain boundary are controlled through boundary conditions. Solution of the governing-conservation equations in time requires discretization of the time domain. The method of Patankar (1980) is implicit using backward Euler time differencing. The expressions that result from discretizing the governing equations are nonlinear algebraic equations.

The STORM simulator uses a Cartesian grid system. Coordinate directions for Cartesian systems follow the right-hand rule from the x to y to z directions. Positive and negative directions along the x, y, and z coordinates are referred to as East, West, North, South, Top, and Bottom, respectively. The gravitational vector can be aligned arbitrarily with respect to the Cartesian directions.

The system of algebraic equations that include the discretized governing-conservation equations and the constitutive functions are nonlinear. Non-linearities in the soil-moisture retention functions, relative permeability functions, and physical properties near phase transitions are the primary contributors. Conversion of the algebraic equations from nonlinear to linear form follows the iterative Newton-Raphson (Kreyszig 1979) technique for multiple variables. Typically, the technique yields quadratic convergence of the residuals, given sufficiently close estimates of the primary unknowns. Each iteration loop requires the solution of a system of linear equations in terms of the equation residuals. Because only orthogonal grid systems are considered, the system of linear equations has a block-banded structure.

The governing-conservation equations are discretized following the integrated finite difference of Patankar (1980) that is locally and globally conserving. Mass conservation equations for water and air components are nearly identical in form and, therefore, result in similar algebraic forms. The conservation equation for energy differs from the mass conservation equations having diffusive-dispersive and advective components. Discretization of combined diffusive and advective transport requires donor-cell weighting of the transport components, yielding different algebraic forms than those for the mass conservation equations. The conservation equation for solute transport is similar in form to the energy conservation equation, but its discretization uses a different donor-cell weighting scheme, resulting in a separate algebraic form.

3.2 Water and Air Mass Conservation Equation

The water and air mass conservation equations, shown in Equations (2.1) and (2.4), are discretized by assuming a piecewise profile to express the variation in primary variables between node points and integrating over the node volume. The accumulation terms (that is, left-hand-side terms) are integrated over the node volume according to Equation (3.1), where the specific terms for the water and air conservation equations appear as shown in Equations (3.2) and (3.3), respectively. Intrinsic properties for the node volume are represented by properties at the node centroid. Source terms are similarly integrated over the node volume according to Equation (3.4).

$$\frac{\partial}{\partial t}\left[\int_{\overline{V}} \overline{M^j}\, d\overline{V}\right] = \frac{\partial}{\partial t}\left[\overline{M^j}\,\overline{V}\right] \quad \text{for } j = w, a \tag{3.1}$$

$$\frac{\partial}{\partial t}\left[\overline{M^w}\,\overline{V}\right] = \frac{\partial}{\partial t}\left[\overline{V} \sum_{\gamma=l,g} \left(\phi_D \omega_\gamma^w \rho_\gamma s_\gamma\right)\right] \tag{3.2}$$

$$\frac{\partial}{\partial t}\left[\overline{M^a}\,\overline{V}\right] = \frac{\partial}{\partial t}\left[\overline{V} \sum_{\gamma=l,g} \left(\phi_D \omega_\gamma^a \rho_\gamma s_\gamma\right)\right] \tag{3.3}$$

$$\int_{\overline{V}} \dot{m}^j\, d\overline{V} = \dot{m}^j\,\overline{V} \quad \text{for } j = w, a \tag{3.4}$$

Flux terms are evaluated on the node surfaces and for the mass conservation equations comprise advective and diffusive components. Integration of the flux terms over the node volume proceeds by first converting the volumetric integral of flux over a control volume to a surface integral through Green's theorem (Kreyszig 1979) according to Equation (3.5). Discretizing the control volume surfaces into node surfaces and defining flux directions parallel to the surface normal allows the surface integrals to be converted to summations over all node surfaces, according to Equation (3.6). This transformation strictly requires an orthogonal grid system for the flux directions to be aligned with the surface normals. Nonorthogonal systems will yield mass balance errors. Darcy fluxes are discretized, in the six coordinated directions, using upwind interfacial averaging for the component mass fraction, phase density, and relative permeability and harmonic averaging for the intrinsic permeability and phase viscosity according to Equation (3.6). These default interfacial averaging schemes can be altered by user input. Diffusive fluxes are discretized, in the six coordinate directions, using harmonic averaging for the combination of terms that comprise an effective diffusion coefficient. The user can also alter the default interfacial averaging scheme for each parameter. Geometric variables for a Cartesian coordinate system are shown in Figure 3.1 for the X-Z coordinate plane and in Figure 3.2 for the Y-Z coordinate plane.

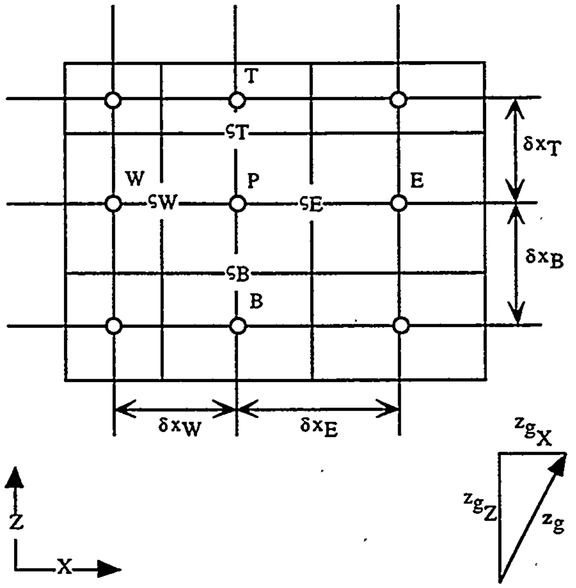

Figure 3.1. X-Z Coordinate Plane for Cartesian Systems

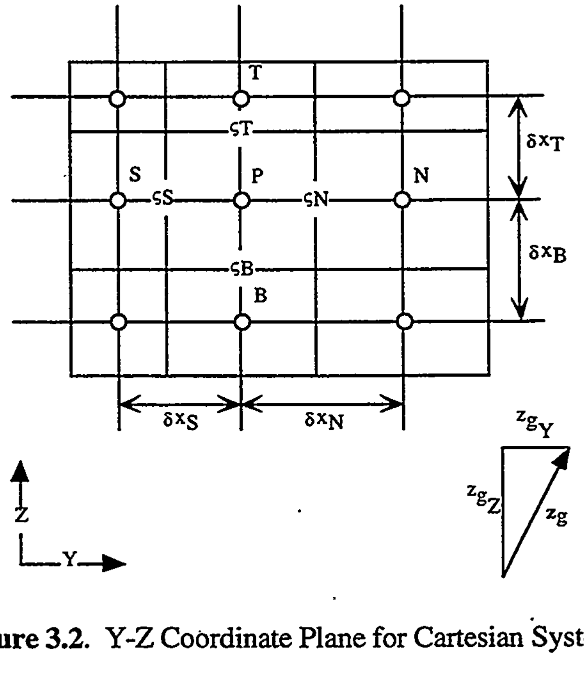

Figure 3.2. Y-Z Coordinate Plane for Cartesian Systems

$$\int_{\overline{V}}\left[\sum_{\gamma=l,g}\left(\nabla\mathbf{F}_{\gamma}^{j}+\nabla\mathbf{J}_{\gamma}^{j}\right)d\overline{V}\right]=\int_{\Gamma}\left[\sum_{\gamma=l,g}\left(\mathbf{F}_{\gamma}^{j}+\mathbf{J}_{\gamma}^{j}\right)\cdot\mathbf{n}\,d\Gamma\right]\ \text{for}\ j=w,a \tag{3.5}$$

where $\int_{\overline{V}}[\ ]d\overline{V}$ = volume integral

$\int_{\Gamma}[\ ]d\Gamma$ = surface integral

$\sum_{\gamma=l,g}$ = summation over aqueous and gas phases

$\sum_{j=w,a}$ = summation over water and air components

where

$$\mathbf{F}_{\gamma}^{j}=\frac{\omega_{\gamma}^{j}\rho_{\gamma}k_{r\gamma}\mathbf{k}}{\mu_{\gamma}}\left(\nabla P_{\gamma}+\rho_{\gamma}g\mathbf{z}_{g}\right)\ \text{for}\ \gamma=l,g\ \text{and}\ j=w,a$$

$$\mathbf{J}_{\gamma}^{j}=-\tau_{\gamma}\phi_{D}\rho_{\gamma}s_{\gamma}\frac{M^{j}}{M_{\gamma}}D_{\gamma}^{j}\nabla\chi_{\gamma}^{j}\ \text{for}\ \gamma=l,g\ \text{and}\ j=w,a$$

$$\int_{\Gamma}\left[\sum_{\gamma=l,g}\left(\mathbf{F}_{\gamma}^{j}+\mathbf{J}_{\gamma}^{j}\right)\cdot\mathbf{n}\,d\Gamma\right]=\sum_{\varsigma=\mathrm{W,E,S,N,T,B}}\left[\sum_{\gamma=\mathrm{l,g}}\left(\mathrm{F}_{\gamma\varsigma}^{\mathrm{j}}+J_{\gamma\varsigma}^{\mathrm{j}}\right)\right]\mathrm{A}_{\varsigma}\ \text{for}\ j=w,a \tag{3.6}$$

where A_{ς} is the area of surface ς, and where,

$$F_{\gamma\varsigma}^{j}=-\frac{\left\langle\omega_{\gamma}^{j}\rho_{\gamma}k_{r\gamma}\right\rangle_{\varsigma}^{uw}\left\langle\mathbf{k}\right\rangle_{\varsigma}^{h}}{\left\langle\mu_{\gamma}\right\rangle_{\varsigma}^{h}}\left[\frac{P_{\gamma\varsigma+}-P_{\gamma\varsigma-}}{\delta x_{\varsigma}}+\left\langle\rho_{\gamma}g\right\rangle_{\varsigma}^{a}z_{g\varsigma}\right]\ \text{for}\ \gamma=l,g;\ j=w,a;\ \text{and}\ \varsigma=E,W,S,N,B,T$$

$$J_{\gamma\varsigma}^{j}=-\left\langle\tau_{\gamma}\phi_{D}\rho_{\gamma}s_{\gamma}\frac{M^{j}}{M_{\gamma}}D_{\gamma}^{j}\right\rangle_{\varsigma}^{h}\frac{\chi_{\gamma\varsigma+}^{j}-\chi_{\gamma\varsigma-}^{j}}{\delta x_{\varsigma}}\ \text{for}\ \gamma=l,g;\ j=w,a;\ \text{and}\ \varsigma=E,W,S,N,B,T$$

and where $\langle\ \rangle_{\varsigma}^{h}$ represents harmonic interfacial averaging at surface ς, $\langle\ \rangle_{\varsigma}^{uw}$ represents upwind or donor cell interfacial averaging at surface ς, $\langle\ \rangle_{\varsigma}^{a}$ represents harmonic interfacial averaging at surface ς, and $\sum_{\varsigma=\mathrm{W,E,S,N,T,B}}$ is the summation over all node surfaces (West, East, South, North, Top, Bottom).

The mass conservation equations are discretized in time using a fully implicit scheme according to Equation (3.7), where the time levels are indicated with superscripts. The primary unknowns for the mass conservation equations are intrinsic properties at node volume centroids (node grid point) for time level $t+\delta t$.

$$\bar{V}\left[\frac{\{\bar{M}^j\}^{t+\delta t} - \{\bar{M}^j\}^{t}}{\delta t}\right] = \bar{V}\{\dot{m}^j\}^{t+\frac{\delta t}{2}} - \sum_{\varsigma=\mathrm{W,E,S,N,T,B}}\left[\sum_{\gamma=l,g}\left(\{\mathrm{F}^{j}_{\gamma\varsigma}\}^{t+\delta t} + \{J^{j}_{\gamma\varsigma}\}^{t+\delta t}\right)\right]\mathrm{A}_\varsigma \quad \text{for } j = w,a \tag{3.7}$$

Converting Equation (3.7) to residual form, where $\bar{R}^j$ is the residual for the mass conservation equation of component j, yields the expression shown in Equation (3.8).

$$\{\bar{R}^j\}^{t+\delta t} = \bar{V}\left[\frac{\{\bar{M}^j\}^{t+\delta t} - \{\bar{M}^j\}^{t}}{\delta t}\right] - \bar{V}\{\dot{m}^j\}^{t+\frac{\delta t}{2}} + \sum_{\varsigma=\mathrm{W,E,S,N,T,B}}\left[\sum_{\gamma=l,g}\left(\{\mathrm{F}^{j}_{\gamma\varsigma}\}^{t+\delta t} + \{J^{j}_{\gamma\varsigma}\}^{t+\delta t}\right)\right]\mathrm{A}_\varsigma \quad \text{for } j = w,a \tag{3.8}$$

3.3 Energy Conservation Equation

The energy conservation equation, shown in Equation (2.7), is discretized by assuming a piecewise profile to express the variation in primary variables between node points and integrating over the node volume. The thermal capacitance terms (that is, left-hand-side terms) and energy source terms are integrated over the node volume according to Equations (3.9) and (3.10), respectively.

$$\frac{\partial}{\partial t}\left[\int_V \bar{E}\,d\bar{V}\right] = \frac{\partial}{\partial t}\left[\bar{E}\bar{V}\right] \quad . \tag{3.9}$$

where

$$\bar{E} = \sum_{\gamma=\ell,g}\left(\phi_D\, \rho_\gamma\, s_\gamma\, u_\gamma\right) + (1-\phi_T)\,\rho_s\, u_s + (\phi_T-\phi_D)\,\rho_\ell\, u_\ell$$

$$\int_{\bar{V}}\left[\sum_{j=w,a}\left(h^j\,\dot{m}^j\right)+\dot{Q}\right]d\bar{V}=\left[\sum_{j=w,a}\left(h^j\,\dot{m}^j\right)+\dot{Q}\right]\bar{V} \tag{3.10}$$

Thermal energy is transferred through advection and diffusion. Spatial discretization of the advective and diffusive thermal flux proceeds by first converting the volumetric integral of flux over a control volume to a surface integral using Green's theorem (Kreyszig 1979), according to Equation (3.11). Surface integrals are approximated by discretizing the control volume surfaces into node surfaces and summing the contributions to heat flux over the node surfaces, according to Equation (3.12).

$$\begin{aligned}&\int_{\bar{V}}\left[-\sum_{\gamma=l,g}\left(\nabla\rho_\gamma h_\gamma \mathbf{q}_\gamma+\sum_{j=w,a}\left(\nabla h^j \mathbf{J}_\gamma^j\right)+\nabla\left(\mathbf{k}_e\nabla T\right)\right)\right]d\bar{V}\\&=\int_{\Gamma}\left[-\sum_{\gamma=l,g}\left(\rho_\gamma h_\gamma \mathbf{q}_\gamma+\sum_{j=w,a}\left(h^j \mathbf{J}_\gamma^j\right)+\left(\mathbf{k}_e\nabla T\right)\right)\right]\mathbf{n}\,d\Gamma\end{aligned} \tag{3.11}$$

where

$$\mathbf{q}_\gamma=-\frac{k_{r\gamma}\mathbf{k}}{\mu_\gamma}\left(\nabla P_\gamma+\rho_\gamma g\mathbf{z}_g\right)\ \text{ for } \gamma=l,g$$

$$\mathbf{J}_\gamma^j=-\tau_\gamma\phi_D\rho_\gamma s_\gamma\frac{M^j}{M_\gamma}D_\gamma^j\nabla\chi_\gamma^j\ \text{ for } \gamma=l,g \text{ and } j=w,a$$

$$\begin{aligned}&\int_{\Gamma}\left[-\sum_{\gamma=l,g}\left(\rho_\gamma h_\gamma \mathbf{q}_\gamma+\sum_{j=w,a}\left(h^j \mathbf{J}_\gamma^j\right)+\left(\mathbf{k}_e\nabla T\right)\right)\right]\mathbf{n}\,d\Gamma\\&=\sum_{\varsigma=W,E,S,N,B,T}\left[-\sum_{\gamma=l,g}\left(\left(\rho_\gamma h_\gamma \mathbf{q}_\gamma\right)_\varsigma+\sum_{j=w,a}\left(h^j \mathbf{J}_\gamma^j\right)_\varsigma+\left(\mathbf{k}_e\nabla T\right)_\varsigma\right)\right]A_\varsigma\end{aligned} \tag{3.12}$$

where

$$q^j_{\gamma\varsigma} = -\frac{\langle k_{r\gamma}\rangle_\varsigma^{uw} \langle \mathbf{k}\rangle_\varsigma^{h}}{\langle \mu_\gamma\rangle_\varsigma^{h}} \left[\frac{P_{\gamma\varsigma+} - P_{\gamma\varsigma-}}{\delta x_\varsigma} + \langle \rho_\gamma g\rangle_\varsigma^{a} z_{g\varsigma} \right] \text{ for } \gamma = l,g; j = w,a; \text{ and } \varsigma = E,W,S,N,B,T$$

$$J^j_{\gamma\varsigma} = -\left\langle \tau_\gamma \phi_D \rho_\gamma s_\gamma \frac{M^j}{M_\gamma} D^j_\gamma \right\rangle_\varsigma^{h} \frac{\chi^j_{\gamma\varsigma+} - \chi^j_{\gamma\varsigma-}}{\delta x_\varsigma} \text{ for } \gamma = l,g; j = w,a; \text{ and } \varsigma = E,W,S,N,B,T$$

The diffusive term of Equation (3.12) is computed using a user-defined interfacial average for the effective thermal conductivity, where the default form is harmonic averaging, according to Equation (3.13). The advective components of Equation (3.12) are computed using upwind (donor cell) averaging, according to Equations (3.14) and (3.15).

$$(\mathbf{k}_e \nabla T)_\varsigma = \langle \mathbf{k}_e \rangle_\varsigma^{h} \frac{T_{\varsigma+} - T_{\varsigma-}}{\delta x_\varsigma} \text{ for } \varsigma = E,W,S,N,B,T \qquad (3.13)$$

$$\begin{gathered}(\rho_\gamma h_\gamma \mathbf{q}_\gamma)_\varsigma = (\rho_\gamma h_\gamma)_{\varsigma-} \max\left[q_{\gamma\varsigma-}, 0\right] - (\rho_\gamma h_\gamma)_{\varsigma+} \max\left[-q_{\gamma\varsigma+}, 0\right] \\ \text{for } \gamma = l,g \text{ and } \varsigma = E,W,S,N,B,T\end{gathered} \qquad (3.14)$$

$$\begin{gathered}(h^j \mathbf{J}^j_\gamma)_\varsigma = (h^j)_{\varsigma-} \max\left[J^j_{\gamma-}, 0\right] - (h^j)_{\varsigma+} \max\left[-J^j_{\gamma+}, 0\right] \\ \text{for } \gamma = l,g \text{ and } \varsigma = E,W,S,N,B,T\end{gathered} \qquad (3.15)$$

As with the mass conservation equations, the energy conservation equations are discretized in time using a fully implicit scheme, according to Equation (3.16), where the time levels are indicated with superscripts. The primary unknowns for the energy equation are the temperatures at the node centroids (node grid points) for time level t + δt. In residual form, Equation (3.16) appears as shown in Equation (3.17).

$$\begin{aligned}\bar{V}\left[\frac{\{\bar{E}\}^{t+\delta t} - \{\bar{E}\}^{t}}{\delta t}\right] &= \bar{V}\left\{\sum_{j=w,a}\left(h^j \dot{m}^j\right) + \dot{Q}\right\}^{t+\frac{\delta t}{2}} = \sum_{\varsigma=W,E,S,N,B,T}\left[\left\{(\mathbf{k}_e \nabla T)_\varsigma\right\}^{t+\delta t}\right] A_\varsigma \\ &- \sum_{\varsigma=W,E,S,N,B,T}\left[\sum_{\gamma=l,g}\left(\left\{(\rho_\gamma h_\gamma \mathbf{q}_\gamma)_\varsigma\right\}^{t+\delta t} - \sum_{j=w,a}\left\{(h^j \mathbf{J}^j_\gamma)_\varsigma\right\}^{t+\delta t}\right)\right] A_\varsigma\end{aligned} \qquad (3.16)$$

$$
\begin{aligned}
\left\{\overline{R}_e\right\}^{t+\delta t} &= \overline{V}\left[\frac{\left\{\overline{E}\right\}^{t+\delta t} - \left\{\overline{E}\right\}^{t}}{\delta t}\right] - \overline{V}\left\{\sum_{j=w,a}\left(h^j \dot{m}^j\right) + \dot{Q}\right\}^{t+\frac{\delta t}{2}} \\
&- \sum_{\varsigma=W,E,S,N,B,T}\left[\left\{\left(\mathbf{k}_e \nabla T\right)_\varsigma\right\}^{t+\delta t}\right] A_\varsigma \\
&+ \sum_{\varsigma=W,E,S,N,B,T}\left[\sum_{\gamma=l,g}\left(\left\{\left(\rho_\gamma h_\gamma \mathbf{q}_\gamma\right)_\varsigma\right\}^{t+\delta t} - \sum_{j=w,a}\left\{\left(h^j \mathbf{J}_\gamma^j\right)_\varsigma\right\}^{t+\delta t}\right)\right] A_\varsigma
\end{aligned}
\tag{3.17}
$$

3.4 Solute Conservation Equation

The solute mass conservation equation, shown in Equation (2.10) is discretized by assuming a piecewise profile for the solute concentration between node points and integrating over the node volume. The advection and diffusion-dispersion transport terms of the solute mass conservation equation are combined following the power-law scheme of Patankar (1980). Integration of the accumulation, source, and decay terms for solute mass over the node volume proceeds according to Equation (3.18).

$$
\int_{\overline{V}}\left(\frac{\partial C}{\partial t} - \dot{m}^C - \dot{R}^C C\right) d\overline{V} = \frac{\partial C}{\partial t}\overline{V} - \dot{m}^C \overline{V} - \dot{R}^C C \overline{V}
\tag{3.18}
$$

Solute transport fluxes are computed between node points and comprise advective and diffusive-dispersive components. Integration of the flux terms proceeds by converting volumetric integrals to surface integrals following Green's theorem (Kreyszig 1979), according to Equation (3.19). Surface integrals are approximated by discretizing the control volume surfaces into node surfaces and summing the contributions to solute transport over the node surfaces, according to Equation (3.20). The diffusive-dispersive term of Equation (3.20) is computed using a user-defined interfacial average where the default scheme is harmonic averaging, according to Equation (3.21).

$$
\begin{aligned}
&\int_{\overline{V}}\left[-\sum_{\gamma=l,g}\left(\nabla\left[C_\gamma q_\gamma\right]\right) + \sum_{\gamma=l,g}\left(\nabla\left[\left(\tau_\gamma\, s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma}\right)\nabla C_\gamma\right]\right)\right] d\overline{V} \\
&= \int_{\Gamma}\left[-\sum_{\gamma=l,g}\left(C_\gamma q_\gamma\right) + \sum_{\gamma=l,g}\left(\left(\tau_\gamma\, s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma}\right)\nabla C_\gamma\right)\right] \cdot \mathbf{n}\, d\Gamma
\end{aligned}
\tag{3.19}
$$

where

$$\mathbf{q}_\gamma = -\frac{k_{r\gamma}\,\mathbf{k}}{\mu_\gamma}\left(\nabla P_\gamma + \rho_\gamma\, g\, \mathbf{z}_g\right) \mathit{for}\ \gamma = \ell, g$$

$$\begin{aligned} &\int_\Gamma \left[-\sum_{\gamma=l,g} \left(C_\gamma q_\gamma\right) + \sum_{\gamma=l,g} \left(\left(\tau_\gamma\ s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma}\right) \nabla C_\gamma \right) \right] \cdot \mathbf{n}\, d\Gamma \\ &= \sum_{\varsigma=W,E,S,N,B,T} \left[-\sum_{\gamma=l,g} \left(C_\gamma q_\gamma\right)_\varsigma + \sum_{\gamma=l,g} \left(\left(\tau_\gamma\ s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma}\right) \nabla C_\gamma \right)_\varsigma \right] A_\varsigma \end{aligned} \tag{3.20}$$

where

$$q_{\gamma\varsigma}^{j} = -\frac{\left\langle k_{r\gamma}\right\rangle_\varsigma^{uw} \left\langle \mathbf{k}\right\rangle_\varsigma^{h}}{\left\langle \mu_\gamma\right\rangle_\varsigma^{h}} \left[\frac{P_{\gamma\varsigma+} - P_{\gamma\varsigma-}}{\delta x_\varsigma} + \left\langle \rho_\gamma g\right\rangle_\varsigma^{a} z_{g\varsigma} \right] \text{ for } \gamma = l, g; j = w, a; \text{ and } \varsigma = E, W, S, N, B, T$$

$$\left(\left(\tau_\gamma\ s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma}\right) \nabla C_\gamma \right)_\varsigma = \left\langle \tau_\gamma\ s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma} \right\rangle_\varsigma^{h} \frac{C_{\gamma\varsigma+} - C_{\gamma\varsigma-}}{\delta x_\varsigma} \tag{3.21}$$

$$\text{for } \gamma = l, g \text{ and } \varsigma = E, W, S, N, B, T$$

Solution of the solute transport equation depends on the local Peclet number that represents the ratio of advective transport to diffusive-dispersive transport, according to Equation (3.22)

$$Pe_{\gamma_\varsigma} = \frac{q_{\gamma\varsigma}\,\delta x_\varsigma}{\left\langle \tau_\gamma\ s_\gamma \phi_D D_\gamma^C + s_\gamma \phi_D \mathbf{D}_{h\gamma} \right\rangle_\varsigma^{h}} \tag{3.22}$$

The power law scheme is based on the solute concentration profile for steady conditions with no sources or decay. For a Peclet number of zero, diffusion-dispersion transport dominates and a linear profile of solute concentration occurs between two spatial points. For a Peclet number of one, advection and diffusion-dispersion equally contribute to solute transport. The solute concentration profile will be skewed towards an upstream solute concentration. For large Peclet numbers, advection transport dominates and the upstream solute concentration defines the solute concentration profile between two spatial points. The power-law scheme closely approximates the exact solution for steady conditions without excessive computational expense. Solute flux from combined advective and diffusive-dispersive transport can be expressed using the power-law scheme according to Equation (3.23).

$$G^C_{\gamma_\varsigma} = -C_{\gamma_{\varsigma+}} \left[\max\left(-q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1 - \left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma} \right) \right)^5, 0 \right) \right]$$

$$+C_{\gamma_{\varsigma-}} \left[\max\left(q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1 - \left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma} \right) \right)^5, 0 \right) \right] \qquad (3.23)$$

for $\gamma = l, g$ and $\varsigma = W, E, S, N, B, T$

where

$$\left(D^C_{\gamma_e}\right)_\varsigma = \frac{\left\langle \tau_\gamma \, s_\gamma \phi_D D^C_\gamma + s_\gamma \phi_D \mathbf{D}_{h\gamma} \right\rangle^h_\varsigma}{\delta x_\varsigma}$$

The discretized solute concentration equation can be written in two compact forms. The first form uses the expressions of solute flux according to Equation (3.24). The second form uses a linear system format with coefficients for the solute concentrations according to Equation (3.25).

$$\sum_{\gamma=l,g} \left(G^C_{\gamma_W} - G^C_{\gamma_E} + G^C_{\gamma_S} - G^C_{\gamma_N} + G^C_{\gamma_B} - G^C_{\gamma_T} \right) = \frac{\partial C}{\partial t}\overline{V} - \dot{m}^C \overline{V} - \dot{R}^C C \overline{V} \qquad (3.24)$$

$$a_P C_P - \sum_{\varsigma=W,E,S,N,B,T} a_\varsigma C_\varsigma = \frac{\partial C}{\partial t}\overline{V} - \dot{m}^C \overline{V} - \dot{R}^C C \overline{V} \qquad (3.25)$$

where

$$a_\varsigma = \sum_{\gamma=l,g} a_{\gamma_\varsigma}$$

$$a_{\gamma_P} = \sum_{\varsigma=W,E,S,N,B,T} a_{\gamma_\varsigma} + \sum_{\varsigma=E,N,T} q_{\gamma_\varsigma} - \sum_{\varsigma=W,S,B} q_{\gamma_\varsigma}$$

$$a_{\gamma_\varsigma} = \max\left(q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1-\left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma}\right)\right)^5,0\right) \text{ for } \gamma = l,g \text{ and } \varsigma = W,S,B$$

$$a_{\gamma_\varsigma} = \max\left(-q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1-\left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma}\right)\right)^5,0\right) \text{ for } \gamma = l,g \text{ and } \varsigma = E,N,T\,.$$

The solute mass conservation equation is discretized in time using a fully implicit scheme, according to Equation (3.26), in linear system format, where time levels are indicated with superscripts.

$$\sum_{\varsigma=P,w,E,S,N,B,T} a_\varsigma \left\{C_\varsigma\right\}^{t+\delta t} = \frac{\left\{C_P\right\}^t}{\partial t}\bar{V} - \dot{m}^C\bar{V} \tag{3.26}$$

where

$$a_\varsigma = \sum_{\gamma=l,g} a_{\gamma_\varsigma}$$

$$a_{\gamma_P} = \sum_{\varsigma=W,E,S,N,B,T} a_{\gamma_\varsigma} + \sum_{\varsigma=E,N,T} V_{\gamma_\varsigma} - \sum_{\varsigma=W,S,B} V_{\gamma_\varsigma} + \frac{\bar{V}}{\partial t} + \dot{R}^C\bar{V}$$

$$a_{\gamma_\varsigma} = \max\left(q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1-\left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma}\right)\right)^5,0\right) \text{ for } \gamma = l,g \text{ and } \varsigma = W,S,B$$

$$a_{\gamma_\varsigma} = \max\left(-q_{\gamma_\varsigma},0\right) + \left(D^C_{\gamma_e}\right)_\varsigma \max\left(\left(1-\left(\frac{0.1\left|q_{\gamma_\varsigma}\right|}{\left(D^C_{\gamma_e}\right)_\varsigma}\right)\right)^5,0\right) \text{ for } \gamma = l,g \text{ and } \varsigma = E,N,T\,.$$

3.4.1 Newton-Raphson Linearization

The discretized governing equations for component mass conservation and energy conservation, Equations (3.8) and (3.17), form a nonlinear set of algebraic equations. Nonlinear equation sets arise from the dependence of secondary variables on the primary unknowns. Cross dependencies of secondary variables in one governing equation with the primary variable unknown in other governing equations requires this system of nonlinear equations be solved simultaneously. A primary assumption associated with the solute mass conservation equation is that solute concentrations are infinitely dilute. This assumption implies that fluid properties are independent of solute concentrations, which allows the solute conservation equations to be solved separately from the coupled flow and energy transport solutions. The discretized governing equations for solute mass conservation, Equation (3.26), form a linear set of algebraic equations that are solved directly following the solution of the coupled flow and energy transport system. The nonlinear situations in the coupled flow and energy transport system of equations are resolved through the application of the iterative Newton-Raphson technique.

The Newton-Raphson linearization technique is an iterative method for solving nonlinear algebraic equations of the form shown in Equation (3.27), where $f(x)$ is differentiable in x. The linearization concept approximates $f(x)$ with suitable tangents. Each iteration yields a new estimate of x as the intersection of the tangent to the function $f(x)$ at the previous estimate of x and the abscissa axis, according to Equation (3.28) in mathematical form. In this formulation, $f(x)$ is considered the equation residual. For convergent systems, the residual decreases quadratically with iteration. In multiple variable systems, as with the coupled flow and energy transport system, the scalar function, $f(x)$, is replaced with a vector function $\mathbf{R}(\mathbf{x})$, according to Equation (3.29). The vector function, $\mathbf{R}(\mathbf{x})$, represents the system of nonlinear algebraic equations produced from discretizing the conservation equations for component mass and energy, Equations (3.8) and (3.17). The vector of unknowns, $\mathbf{x}$, represents the set of primary variables for the system that are determined by the operational mode and phase conditions. Equation (3.29) can be rewritten in terms of increments to the primary variables, according to Equation (3.30). The partial derivatives shown in Equation (3.30) form the Jacobian matrix.

$$f(x)=0 \tag{3.27}$$

$$x^{n+1}=x^{n}-\frac{f(x)}{f'(x)} \tag{3.28}$$

$$x^{n+1}=x^{n}-\frac{\mathbf{R}(\mathbf{x})}{\mathbf{R}'(\mathbf{x})} \tag{3.29}$$

$$\frac{\partial \mathbf{R}(\mathbf{x})}{\partial \mathbf{x}}\Delta\mathbf{x}=-\mathbf{R}(\mathbf{x}) \tag{3.30}$$

To simplify these discussions, a one-dimensional system involving the solution of the water mass, air mass, and energy conservation equations is considered. The system of linear equations that result from applying the Newton-Raphson linearization technique to this system of nonlinear algebraic equations is shown in Equation (3.31) for a computational domain with n nodes. In Equation (3.31), each Jacobian matrix element represents a block matrix of order three, according to Equation (3.32). Each unknown vector element represents a vector of increments to the primary variables of order three, according to Equation (3.33). Each solution vector element represents a vector of equation residuals of order three, according to Equation (3.34).

$$\begin{vmatrix}
\frac{\partial \mathbf{R}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{x}_2} & 0 & 0 & \cdot & 0 & 0 & 0 \\
\frac{\partial \mathbf{R}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{x}_3} & 0 & \cdot & 0 & 0 & 0 \\
0 & \frac{\partial \mathbf{R}_3}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{x}_4} & \cdot & 0 & 0 & 0 \\
0 & 0 & \frac{\partial \mathbf{R}_4}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{R}_4}{\partial \mathbf{x}_4} & \cdot & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & 0 & 0 & 0 & \cdot & \frac{\partial \mathbf{R}_{n-2}}{\partial \mathbf{x}_{n-2}} & \frac{\partial \mathbf{R}_{n-2}}{\partial \mathbf{x}_{n-1}} & 0 \\
0 & 0 & 0 & 0 & \cdot & \frac{\partial \mathbf{R}_{n-1}}{\partial \mathbf{x}_{n-2}} & \frac{\partial \mathbf{R}_{n-1}}{\partial \mathbf{x}_{n-1}} & \frac{\partial \mathbf{R}_{n-1}}{\partial \mathbf{x}_{n}} \\
0 & 0 & 0 & 0 & \cdot & 0 & \frac{\partial \mathbf{R}_{n}}{\partial \mathbf{x}_{n-1}} & \frac{\partial \mathbf{R}_{n}}{\partial \mathbf{x}_{n}}
\end{vmatrix}
\begin{vmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \cdot \\ \mathbf{x}_{n-2} \\ \mathbf{x}_{n-1} \\ \mathbf{x}_n \end{vmatrix}
=
\begin{vmatrix} -\mathbf{R}_1 \\ -\mathbf{R}_2 \\ -\mathbf{R}_3 \\ -\mathbf{R}_4 \\ \cdot \\ -\mathbf{R}_{n-2} \\ -\mathbf{R}_{n-1} \\ -\mathbf{R}_n \end{vmatrix}
\qquad (3.31)$$

$$\frac{\partial \mathbf{R}_k}{\partial \mathbf{x}_l} =
\begin{vmatrix}
\frac{\partial R_k^w}{\partial PV_l^w} & \frac{\partial R_k^w}{\partial PV_l^a} & \frac{\partial R_k^w}{\partial PV_l^e} \\
\frac{\partial R_k^a}{\partial PV_l^w} & \frac{\partial R_k^a}{\partial PV_l^a} & \frac{\partial R_k^a}{\partial PV_l^e} \\
\frac{\partial R_k^e}{\partial PV_l^w} & \frac{\partial R_k^e}{\partial PV_l^a} & \frac{\partial R_k^e}{\partial PV_l^e}
\end{vmatrix}
\qquad (3.32)$$

$$-\mathbf{x}_l = \begin{vmatrix} \Delta PV_l^w \\ \Delta PV_l^a \\ \Delta PV_l^e \end{vmatrix} \qquad (3.33)$$

$$-\mathbf{R}_k = \begin{vmatrix} -R_k^w \\ -R_k^a \\ -R_k^e \end{vmatrix} \tag{3.34}$$

For a 2-dimensional system involving the solution of the water mass, air mass, and energy conservation equations, the Jacobian matrix contains two extra bands of block matrixes: one below and one above the diagonal band. These extra bands are located one half-band width from the main diagonal band, where the half-band width equaled the lesser of the number of nodes per row or column for a 2-dimensional grid. A 3-dimensional grid contains four extra bands of block matrixes, two below and two above the diagonal band. The furthest bands are located one half-band width from the main diagonal band, where the half-band width equaled the least number of nodes in a plane. For example, a 3-dimensional Cartesian grid with 20 nodes in the X coordinate direction, 30 nodes in the Y coordinate direction, and 40 nodes in the Z coordinate direction, has a half-band width of 600.

The Newton-Raphson scheme has two major computational components. The first component involves computing the Jacobian matrix and solution vector elements; the second component involves solving the resulting linear system of equations. During each iteration, both components are required. The Jacobian matrix elements are essentially partial derivatives of the equation residuals with respect to the primary variables. These partial derivatives are computed numerically as exemplified in Equation (3.35)

$$\frac{\partial R_k^j}{\partial PV_l^i} \cong \frac{R_k^j\Big|^{PV_l^i+\Delta PV_l^i} - R_k^j\Big|^{PV_l^i}}{\Delta PV_l^i} . \tag{3.35}$$

For robustness and directness in coding, numerical evaluation of these partial derivatives was chosen over analytical evaluation. The numerical evaluation scheme requires that the equation residuals be evaluated with the most current values of the primary variables and with each primary variable slightly incremented. Increments to the primary variables are critical parameters for implementing the Newton-Raphson linearization scheme using numerical derivatives. Excessively large increments can lead to non-convergent solutions because the tangents to the residual equations are computed incorrectly. In contrast, excessively small increments can result in changes in the equation residuals that are below the precision limit.

Each Newton-Raphson iteration procedure begins with the start of a new time step or after a convergence failure and reduction in time step. Each iteration begins by executing logic algorithms that determine the primary variable set from phase conditions and fix the primary variable increments. The next step is to calculate all secondary variables with the current primary variables and with each primary variable incremented. Following this, all flux variables are evaluated with the current primary variable and associated secondary variables and then repeated with the incremented primary and associated secondary variables. With the secondary variables and fluxes evaluated, the residuals to the governing equations are computed using current and incremented values of the primary, secondary, and flux variables. The current and incremented equations residuals are then used to compute the partial derivatives that comprise the Jacobian matrix and the solution vector.

The resulting system of linear equations is then solved with either a direct or iterative linear system solver, with the unknowns as increments to the primary variables. Converged solutions are recognized by comparing the maximum normalized increment to the primary variables against a convergence criterion (that is, a user-specified limit). A Newton-Raphson iteration ends by updating the primary variables with the latest computed primary increments. If the largest normalized primary variable increment exceeds the convergence limit, another iteration is executed. If the solution is convergent, the solute transport equations are solved or a new time step is initiated. In the case of a non-convergent solution, the time step is reduced, the primary variables are reset to the most recent converged values, and the time step is reinitiated.

3.4.2 Linear System Solvers

The system of linear equations that results from the discretization of the governing partial differential equations over a particular computational domain always has a banded structure, where the number of off diagonal bands equals the dimensions of the computational domain. Elements within the Jacobian coefficient are actually submatrixes, where the submatrix order equals the number of solved coupled governing equations. For example, the solution of only the water conservation equation produces single element submatrixes, whereas the simultaneous solution of the three mass conservation equations with the energy equation yields four-by-four submatrixes. These structured Jacobian coefficient matrixes arise from the element sequencing scheme incorporated into the STORM simulator and the limitations on the computational domain structures. The matrix sequencing or numbering algorithms are designed to minimize the largest half-band width for a given computational domain and solution option combination. The sequencing algorithms are designed for orthogonal grid systems that are six-surfaced in three dimensions or four-sided in two dimensions.

Two linear equation solvers are available within the STORM simulator, a direct-banded matrix algorithm and an iterative conjugate gradient algorithm. The banded matrix algorithm is generally more appropriate for small to moderately sized Jacobian matrixes (order less than 35,000); whereas, the conjugate gradient algorithm appears more appropriate for larger order Jacobian matrixes. In general, the banded matrix algorithm requires more memory than the conjugate gradient algorithm that uses an efficient sparse matrix storage scheme. The banded matrix algorithm is computationally more efficient on small to moderately sized problems; however, for larger problems the conjugate gradient algorithm is the better performer. Both linear equation solution algorithms were obtained from publicly available software packages.

The banded matrix solution algorithm was extracted from the LINPACK subroutines (Dongarra et al. 1979) for general nonsymmetric band matrixes. The algorithm operates on band matrixes by decomposing the matrix into an upper triangular and lower triangular matrix. The matrix product of the lower triangular matrix times the upper triangular matrix equals the original band matrix (that is, $\mathbf{A} = \mathbf{L} \cdot \mathbf{U}$, where $\mathbf{A}$ is the band matrix, $\mathbf{L}$ is the lower triangular matrix and $\mathbf{U}$ is the upper triangular matrix). The system of linear equations, $\mathbf{A}\cdot\mathbf{x}=\mathbf{b}$, is solved with the preceding decomposition or factorization by solving successively $\mathbf{L}(\mathbf{U}\cdot\mathbf{x})=\mathbf{b}$. This factorization procedure produces nonzero elements outside the bands of the original band matrix. If m_l equals the half-band width of the Jacobian coefficient matrix (the STORM simulator produces band matrixes with equal lower and upper bandwidths), the two triangular factors have bandwidths of m_l and $2m_l$. Storage must be provided for the extra m_l diagonals. This requirement is illustrated for a 1-dimensional problem of five nodes and two solved mass conservation equations. The Jacobian coefficient matrix for this problem appears as shown in Equation (3.36). The band storage requires $3m_l + 1 = 10$ rows of storage arranged as shown in Equation (3.37). The * indicates elements that are never referenced but for which storage space must be provided. The + indicates elements that may be occupied during the factorization process. The original Jacobian coefficient matrix is referred to as **A** and its storage counterpart as **a**. The columns of **A** are stored in the columns of **a**, and the diagonals of **A** are stored in the rows of **a**, so the principal diagonal is stored in row $2m_l + 1$ of **a**.

$$\mathbf{A}=\begin{vmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & 0 & 0 & 0 & 0 & 0 & 0 \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & 0 & 0 & 0 & 0 & 0 & 0 \\ x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & x_{3,5} & x_{3,6} & 0 & 0 & 0 & 0 \\ x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & x_{4,5} & x_{4,6} & 0 & 0 & 0 & 0 \\ 0 & 0 & x_{5,3} & x_{5,4} & x_{5,5} & x_{5,6} & x_{5,7} & x_{5,8} & 0 & 0 \\ 0 & 0 & x_{6,3} & x_{6,4} & x_{6,5} & x_{6,6} & x_{6,7} & x_{6,8} & 0 & 0 \\ 0 & 0 & 0 & 0 & x_{7,5} & x_{7,6} & x_{7,7} & x_{7,8} & x_{7,9} & x_{7,10} \\ 0 & 0 & 0 & 0 & x_{8,5} & x_{8,6} & x_{8,7} & x_{8,8} & x_{8,9} & x_{8,10} \\ 0 & 0 & 0 & 0 & 0 & 0 & x_{9,7} & x_{9,8} & x_{9,9} & x_{9,10} \\ 0 & 0 & 0 & 0 & 0 & 0 & x_{10,7} & x_{10,8} & x_{10,9} & x_{10,10} \end{vmatrix} \qquad (3.36)$$

$$\mathbf{a}=\begin{vmatrix} * & * & * & * & * & * & + & + & + & + \\ * & * & * & * & * & + & + & + & + & + \\ * & * & * & * & + & + & + & + & + & + \\ * & * & * & x_{1,4} & 0 & x_{3,6} & 0 & x_{5,8} & 0 & x_{7,10} \\ * & * & x_{1,3} & x_{2,4} & x_{3,5} & x_{4,6} & x_{5,7} & x_{6,8} & x_{7,9} & x_{8,10} \\ * & x_{1,2} & x_{2,3} & x_{3,4} & x_{4,5} & x_{5,6} & x_{6,7} & x_{7,8} & x_{8,9} & x_{9,10} \\ x_{1,1} & x_{2,2} & x_{3,3} & x_{4,4} & x_{5,5} & x_{6,6} & x_{7,7} & x_{8,8} & x_{9,9} & x_{10,10} \\ x_{2,1} & x_{3,2} & x_{4,3} & x_{5,4} & x_{6,5} & x_{7,6} & x_{8,7} & x_{9,8} & x_{10,9} & * \\ x_{3,1} & x_{4,2} & x_{5,3} & x_{6,4} & x_{7,5} & x_{8,6} & x_{9,7} & x_{10,8} & * & * \\ x_{4,1} & 0 & x_{6,3} & 0 & x_{8,5} & 0 & x_{10,7} & * & * & * \end{vmatrix} \qquad (3.37)$$

The conjugate gradient solution algorithm was extracted from a package of subroutines intended for solving large sparse linear systems by iterative methods (Oppe et al. 1988). This package of subroutines is referred to as Nonsymmetric Preconditioned Conjugate Gradient (NSPCG) and has various acceleration techniques and preconditioners available for solving large sparse linear systems. The algorithms available from the NSPCG package are the Incomplete Cholesky (IC) preconditioner and the Generalized Minimal Residual Method (GMRES) accelerator. Preconditioners compute a splitting matrix Q so the preconditioned system of linear equations, $\mathbf{Q}^{-1}\mathbf{A}\mathbf{x}=\mathbf{Q}^{-1}\mathbf{b}$, is better conditioned than the original system, $\mathbf{A}\cdot\mathbf{x}=\mathbf{b}$. The splitting matrix for the IC preconditioner is an incomplete LU decomposition of the Jacobian matrix A. The form of the splitting matrix is $\mathbf{Q}=(\mathbf{D}-\mathbf{S})\mathbf{D}^{-1}(\mathbf{D}-\mathbf{T})$, where D is a diagonal matrix containing the factorization pivots, S is a lower triangular matrix, and T is an upper triangular matrix. It is necessary to store both S and T because the Jacobian coefficient matrix A is nonsymmetrical.

The NSPCG package allows several modes of storage for the Jacobian coefficient matrix. The storage mode selected for the STORM simulator represents the Jacobian coefficient matrix with two rectangular arrays, one real and one integer. Both arrays are dimensioned n by m, where n equals the number of unknowns and m equals seven times the number of solved coupled governing equations. Each row in the real coefficient matrix **C** contains the nonzero values of the corresponding row in the Jacobian coefficient matrix A; the corresponding row in the integer coefficient matrix J contains the column indexes. The Jacobian matrix, shown in Equation (3.36), is expressed as the real and integer matrixes shown in Equations (3.38) and (3.39), respectively. Although this storage format requires two coefficient matrixes, the storage requirements for large sparse systems (such as those for large multidimensional problems) are greatly reduced over those for the banded matrix solution scheme.

$$\mathbf{C}=\begin{vmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & 0 & 0 \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & 0 & 0 \\ x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & x_{3,5} & x_{3,6} \\ x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & x_{4,5} & x_{4,6} \\ x_{5,1} & x_{5,2} & x_{5,3} & x_{5,4} & x_{5,5} & x_{5,6} \\ x_{6,1} & x_{6,2} & x_{6,3} & x_{6,4} & x_{6,5} & x_{6,6} \\ x_{7,1} & x_{7,2} & x_{7,3} & x_{7,4} & x_{7,5} & x_{7,6} \\ x_{8,1} & x_{8,2} & x_{8,3} & x_{8,4} & x_{8,5} & x_{8,6} \\ x_{9,1} & x_{9,2} & x_{9,3} & x_{9,4} & 0 & 0 \\ x_{10,1} & x_{10,2} & x_{10,3} & x_{10,4} & 0 & 0 \end{vmatrix} \qquad (3.38)$$

$$\mathbf{J}=\begin{vmatrix} 1 & 2 & 3 & 4 & 0 & 0 \\ 1 & 2 & 3 & 4 & 0 & 0 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 6 & 7 & 8 & 9 & 10 \\ 5 & 6 & 7 & 8 & 9 & 10 \\ 7 & 8 & 9 & 10 & 0 & 0 \\ 7 & 8 & 9 & 10 & 0 & 0 \end{vmatrix} \tag{3.39}$$

3.5 Geochemical Reactions

Combining Equation (2.17) and Equation (2.20) yields the solute mass conservation equation including geochemical reactions

$$\begin{gathered} \frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K_H^{'-1} + s_\ell\right) C_i^\ell\right] = \\ \nabla \bullet \left[-\phi_D \rho_\ell \left(s_g \mathbf{V}_g K_H^{'-1} + s_\ell \mathbf{V}_\ell\right) C_i^\ell + \phi_D \rho_\ell \left(s_g \mathbf{D}_i^g K_H^{'-1} + s_\ell \mathbf{D}_i^\ell\right) \cdot \nabla C_i^\ell\right] \\ + \sum_{j=1}^{N_e} v_{ij} W_j^e + \sum_{j=1}^{N_k} v_{ij} W_j^k \end{gathered} \tag{3.40}$$

In matrix form, Equation (3.40) can be written as

$$\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K_H^{'-1} + s_\ell\right) \vec{\mathbf{C}}\right] = \vec{\mathbf{L}}\left(\vec{\mathbf{C}}\right) + \mathbf{N}\vec{\mathbf{W}} \tag{3.41}$$

where

$$\vec{\mathbf{L}}\left(\vec{\mathbf{C}}\right) = \left[L_1\left(C_1\right), \ldots, L_{N_s}\left(C_{N_s}\right)\right]^T$$

$$L_i\left(C_i\right) = \nabla \bullet \left[-\phi_D \rho_\ell \left(s_g \mathbf{V}_g K_H^{'-1} + s_\ell \mathbf{V}_\ell\right) C_i^\ell + \phi_D \rho_\ell \left(s_g \mathbf{D}_i^g K_H^{'-1} + s_\ell \mathbf{D}_i^\ell\right) \cdot \nabla C_i^\ell\right]$$

$$\vec{\mathbf{W}} = \left(W_1^e, \ldots W_{N_e}^e, W_1^k, \ldots W_{N_k}^k\right)^T$$

$$\mathbf{N} = \begin{bmatrix} \nu_{11} & \cdots & \nu_{1,N_r} \\ \cdot & \cdots & \cdot \\ \cdot & \cdots & \cdot \\ \cdot & \cdots & \cdot \\ \nu_{N_s,1} & \cdots & \nu_{N_s,N_r} \end{bmatrix}$$

where N_s is the number of species, N_e is the number of equilibrium reactions, and N_k is the number of kinetic reactions.

3.5.1 Linear Transformation of the Reaction-Transport Equation

Ortoleva et al. (1987) pointed out that a subsurface system usually contains reactions of different time scales, and, therefore, "straightforward numerical solution of the solute equations is impractical." To remedy the stiffness of Equation (3.41), a modification is needed to the equation that separates reactions with different time scales. As discussed in Section 2.3.1, geochemical models usually assume some reactions with a short time scale are equilibrium reactions, while assuming others are kinetic reactions. In most cases, separating reactions with different time scales is separating equilibrium reactions from kinetic reactions.

One approach to performing such separations is based on a canonical representation of reactions (Kirkner and Reeves 1988; Lichtner 1985; Reed 1982; Sevougian et al. 1993; Steefel and Lasaga 1994; Yeh and Tripathi 1991). That approach arbitrarily assigns one subset of species as the primary species (or base species) and the others as the secondary species. Reactions are written in canonical form, that is, each reaction involves only one secondary species, but requires several primary species (Lichtner 1985). Consequently, the rate of production/destruction of primary species, due to reversible reactions, is expressed as the weighted summary of secondary species with the number of moles of primary species in the secondary species as the weight. Adding the mass conservation equation of secondary species with that of the primary species leads to a mass conservation equation that does not contain a source/sink term of reversible reactions. It not only separates equilibrium reactions from kinetic ones, but it also reduces the number of nonlinear partial differential equations (PDE) from N_s to $N_s - N_e$.

From the perspective of linear algebra, the canonical representation approach is equivalent to a linear transformation on aqueous species. The original variables-species concentrations are transformed to total concentrations of chemical components, a linear combination of relevant species. This generalization has led toward a more general approach to separating equilibrium from kinetic reactions.

3.5.2 Q-R Decomposition

Look at the reaction matrix $\mathbf{N}$. Assume its columns are linearly independent (that is, the reactions are linearly independent) with the dimension of $N_s \times N_r$. The reactions are arranged so the first N_e

columns represent equilibrium reactions and the rest are kinetic reactions. The independent assumption guarantees $N_r \le N_s$. Matrix algebra states that $\mathbf{N}$ can be decomposed to

$$\mathbf{N}_{N_s \times N_r} = \mathbf{Q}_{N_s \times N_r} \cdot \begin{bmatrix} \mathbf{R}_{N_s \times N_r} \\ 0_{N_r \times (N_s - N_r)} \end{bmatrix} \tag{3.42}$$

(Golub and Van Loan 1983) where $\mathbf{Q}$ is an orthogonal matrix:

$$\mathbf{Q}^T = \mathbf{Q}^{-1} \tag{3.43}$$

and $\mathbf{R}$ is an upper-triangular matrix:

$$\mathbf{R}_{N_R \times N_R} = \begin{bmatrix} r_{11} & \cdot & \cdots & \cdot & \cdot \\ 0 & r_{22} & \cdots & \cdot & \cdot \\ 0 & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & \cdots & 0 & r_{N_R, N_R} \end{bmatrix} \tag{3.44}$$

Multiplying Equation (3.41) by $\mathbf{Q}^T$ gives

$$\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K_H^{-1} + s_\ell\right) \mathbf{Q}^T \bar{\mathbf{C}}\right] = \mathbf{Q}^T \bar{\mathbf{L}}\left(\bar{\mathbf{C}}\right) + \mathbf{R}\bar{\mathbf{W}} \tag{3.45}$$

Denote

$$\mathbf{Q}^T \bar{\mathbf{C}} = \bar{\chi} = \left(\chi_1, \ldots, \chi_{N_e}, \chi_{N_e+1}, \ldots, \chi_{N_e+N_k}, \chi_{N_e+N_k+1}, \ldots, \chi_{N_s}\right)^T \tag{3.46}$$

and recognize that each χ_i is a linear combination of (C_i). Following the notes from the canonical representation approach (Lichtner 1985), χ_i is called the general component. However, χ_i does not bear the straightforward physical meaning as the chemical components do.

Also denote

$$\mathbf{R}\vec{\mathbf{W}} = \vec{\beta} = \left(\beta_1, \ldots, \beta_{N_e}, \beta_{N_e+1}, \ldots, \beta_{N_e+N_k}\right)^T \tag{3.47}$$

and

$$\vec{\mathbf{L}}(\vec{\chi}) = \mathbf{Q}^T \vec{\mathbf{L}}\left(\vec{\mathbf{C}}\right). \tag{3.48}$$

Note the last N_k components of $\vec{\beta}$ contain only the source terms of kinetic reactions.

Now Equation (3.45) can be written in three groups with different source terms. Group I, $\left(\chi_1, \ldots, \chi_{N_e}\right)^T$, contains source terms of equilibrium and kinetic reactions:

$$\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K'^{-1}_H + s_\ell\right)\chi_l\right] = L(\chi_l) + \beta_l, \;\; l = 1, \ldots, N_e \tag{3.49}$$

Group II, kinetic reactive general components, $\left(\chi_{N_e+1}, \ldots, \chi_{N_e+N_k}\right)^T$, contains the source terms only of the kinetic reactions:

$$\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K'^{-1}_H + s_\ell\right)\chi_l\right] = L(\chi_l) + \beta_l, \;\; l = N_e + 1, \ldots, N_r \tag{3.50}$$

where $N_r = N_e + N_k$ is the total number of reactions. Group III, non-reactive general components, $\left(\chi_{N_e+N_k+1}, \ldots, \chi_{N_s}\right)^T$, does not contain the source terms of either type of reactions:

$$\frac{\partial}{\partial t}\left[\phi_D \rho_\ell \left(s_g K'^{-1}_H + s_\ell\right)\chi_l\right] = L(\chi_l), \;\; l = N_{r+1}, \ldots, N_s \tag{3.51}$$

Because Equation (3.49) contains source terms of equilibrium reactions, it is still difficult to directly solve Equations (3.49) through (3.51) together. However, because of the equilibrium assumption made for the N_e reactions Equation (3.49) is disregarded and replaced with the N_e equilibrium conditions

$$\prod_{i=1}^{N_s} (\gamma_i C_i)^{v_{ij}} = K_j^{eq}, \;\; j = 1, \ldots, N_e \tag{3.52}$$

Equations (3.50) through (3.52) are the working equations. They contain N_s unknowns and N_s equations; thus, this system of equations is closed.

Note the working equations contain only N_k nonlinear PDE plus N_e algebra equations and $N_s - N_r$ linear PDE. Compared with the canonical representation approach that yields $N_s - N_e$ nonlinear PDE, this approach yields $N_s - N_e - N_k$ fewer nonlinear PDE. At the extreme case where $N_s = N_e + N_k$, those two approaches yield the same number of nonlinear PDE. Reducing the number of nonlinear PDE can save computing time. The real saving depends on the ratio of N_s and N_r. Observations indicate a speed-up factor of 1.5-3.

Unlike the canonical representation approach, the Q-R decomposition approach does not bear the concepts of primary species and secondary species. Thus, it does not impose the constraint of the canonical form on reactions. Eliminating this constraint frees the user from the trivial, but time-consuming, work of writing every reaction in canonical forms. The user has more time to consider whether or not the chemical model is physically relevant.

3.5.3 Finite Difference Discretization

Equations (3.50) and (3.51) are discretized using the control-volume finite difference method on a rectangular 2-D domain (Strikwerda 1989). For notational convenience in the discretization procedure, the subscript i is no longer used to denote different equations. Rather, as by convention, the subscript is used to denote grid point, and superscript denotes different time slices. Equation (3.50) can be discretized to

$$
\begin{aligned}
&\frac{\left\{\phi_D\rho_\ell\left(s_g K_H^{\prime -1}+s_\ell\right)\chi\right\}_{i,j}^{t+\delta t}-\left\{\phi_D\rho_\ell\left(s_g K_H^{\prime -1}+s_\ell\right)\chi\right\}_{i,j}^{t}}{\Delta t}\\
&=\lambda\left[\frac{\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j}^{t+\delta t}-\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i-1,j}^{t+\delta t}}{x_{i,j}-x_{i-1,j}}\right]\\
&+\lambda\left[\frac{\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j}^{t+\delta t}-\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j-1}^{t+\delta t}}{y_{i,j}-y_{i,j-1}}\right]\\
&+(1-\lambda)\left[\frac{\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j}^{t}-\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i-1,j}^{t}}{x_{i,j}-x_{i-1,j}}\right]\\
&+(1-\lambda)\left[\frac{\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j}^{t}-\left\{\phi_D\rho_\ell\left(s_g\mathbf{V}_g K_H^{\prime -1}+s_\ell\mathbf{V}_\ell\right)\chi\right\}_{i,j-1}^{t}}{y_{i,j}-y_{i,j-1}}\right]\\
&+\lambda\left[\frac{\dfrac{\{\chi\}_{i+1,j}^{t+\delta t}-\{\chi\}_{i,j}^{t+\delta t}}{x_{i+1,j}-x_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i+\frac{1}{2},j}-\dfrac{\{\chi\}_{i-1,j}^{t+\delta t}-\{\chi\}_{i,j}^{t+\delta t}}{x_{i-1,j}-x_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i-\frac{1}{2},j}}{\left(x_{i+1,j}-x_{i-1,j}\right)/2}\right]\\
&+(1-\lambda)\left[\frac{\dfrac{\{\chi\}_{i+1,j}^{t}-\{\chi\}_{i,j}^{t}}{x_{i+1,j}-x_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i+\frac{1}{2},j}-\dfrac{\{\chi\}_{i-1,j}^{t}-\{\chi\}_{i,j}^{t}}{x_{i-1,j}-x_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i-\frac{1}{2},j}}{\left(x_{i+1,j}-x_{i-1,j}\right)/2}\right]\\
&+\lambda\left[\frac{\dfrac{\{\chi\}_{i,j+1}^{t+\delta t}-\{\chi\}_{i,j}^{t+\delta t}}{y_{i,j+1}-y_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i,j+\frac{1}{2}}-\dfrac{\{\chi\}_{i,j-1}^{t+\delta t}-\{\chi\}_{i,j}^{t+\delta t}}{y_{i,j-1}-y_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{\prime -1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i,j-\frac{1}{2}}}{\left(y_{i,j+1}-y_{i,j-1}\right)/2}\right]
\end{aligned}
$$

$$+(1-\lambda)\left[\frac{\frac{\{\chi\}_{i,j+1}^{t}-\{\chi\}_{i,j}^{t}}{y_{i,j+1}-y_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{'-1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i,j+\frac{1}{2}}-\frac{\{\chi\}_{i,j-1}^{t}-\{\chi\}_{i,j}^{t}}{y_{i,j-1}-y_{i,j}}\left\{\phi_D\rho_\ell\left(s_g\mathbf{D}_i^g K_H^{'-1}+s_\ell\mathbf{D}_i^\ell\right)\right\}_{i,j-\frac{1}{2}}}{\left(y_{i,j+1}-y_{i,j-1}\right)/2}\right]$$

$$+\beta_l \qquad (3.53)$$

where. λ is a time-weighting parameter: when $\lambda=1$, it is a fully implicit scheme, $\lambda=0.5$ is the Crank-Nicolson scheme; $\lambda=0$, is an explicit scheme. An upwinding scheme is used; thus,

$$i^{'}=\begin{cases} i+1, \text{ when } \mathbf{V}_{\ell x}>0 \\ i \quad, \text{ when } \mathbf{V}_{\ell x}<0 \end{cases}, \text{and } j^{'}=\begin{cases} j+1, \text{ when } \mathbf{V}_{\ell y}>0 \\ j \quad, \text{ when } \mathbf{V}_{\ell y}<0 \end{cases}$$

Equation (3.51) can be discretized in the same manner. The only difference is that Equation (3.51) does not contain the β term.

The main argument against upwinding is the associated numerical diffusion that smears reaction fronts. However, upwinding has the advantage of being less prone to numerical oscillation. As noted by (Yeh et al. 1993), "one scheme resolving the problems of numerical oscillation tends to yield excessive numerical spreading, whereas the other scheme resolving the problems of numerical spreading tends to create severe numerical oscillation."

3.5.4 Fully Coupled Algorithm

Equation (3.53) is a nonlinear equation that can be solved by Newton-Raphson iteration that first converts Equation (3.53) to a linear equation (Press et al. 1986)

$$\mathbf{A}\vec{\delta}^k=\mathbf{b} \qquad (3.54)$$

where $\mathbf{A}$ is the global Jacobian matrix, $\vec{\delta}^k$ is a correction on the previous solution $\vec{\chi}^k$, and $\mathbf{b}$ is the constant term. After solving Equation (3.54), corrections are made on $\vec{\chi}^k$:

$$\vec{\chi}^{k+1}=\vec{\chi}^k-\vec{\delta}^k \qquad (3.55)$$

The new solution $\vec{\chi}^{k+1}$ gives new values of $\mathbf{A}$ and $\mathbf{b}$ that in turn yield $\vec{\chi}^{k+2}$ ... and the iteration is continued until the solution meets convergence criteria.

The preceding algorithm solves the transport and reaction parts in one step. Such an algorithm is referred to as Fully Coupled, although other researchers have given it different names, such as one-step (Mangold and Tsang 1991), DAE (Yeh and Tripathi 1989), and global implicit (Steefel and Lasaga 1994).

As stated by Yeh and Tripathi (1989), the difficulty of implementing a *Fully Coupled* algorithm is in the huge size of the global Jacobian matrix $\mathbf{A}$. In their 2-dimensional example with 60 x 50 grids and 30 species, the global Jacobian matrix $\mathbf{A}$ has 90,000 x 90,000 = 8,100,000,000 entries. Assuming each entry uses 32 bits (double precision), the matrix requires 2025 MB of memory. It is impractical to store such a large matrix in most computers, and even matrix compression techniques cannot help much. However, the difficulty of storing $\mathbf{A}$ does not necessarily mean it is impractical to solve the equation. Many numerical algorithms used to solve Equation (3.54), for example, iteration schemes (Golub and Van Loan 1983), do not need to generate and store the global Jacobian matrix $\mathbf{A}$ in one piece.

Generally, in iteration algorithms, $\mathbf{A}$ is divided into many submatrixes, so that during the iteration procedure, only one of these submatrixes is needed. A natural way is to divide $\mathbf{A}$ into submatrixes with each submatrix containing the information of one node. The size of the submatrixes is $N_s \times N_s$. During the iteration, it is necessary only to solve a much smaller linear equation:

$$\mathbf{A}_{i,i}\bar{\delta}_i = \mathbf{b}_i - \sum_{i \neq j} \mathbf{A}_{i,j}\bar{\delta}_j \tag{3.56}$$

where $\mathbf{A}_{i,i}$ is the i th submatrix in diagonal, while $\mathbf{A}_{i,j}$ is an off-diagonal submatrix.

This way of dividing $\mathbf{A}$ allows the solution of the problem on the entire domain, node by node. Two classical ways are known to iterate the node-by-node scheme: one is the Jacobi iteration; the other is the Gauss-Seidel iteration (Press et al. 1986). The difference between the Jacobi and the Gauss-Seidel iterations is the latter utilizes the latest updated results of iteration, but the former does not. That difference gives the Gauss-Seidel iteration a factor of two in speed over the Jacobi iteration (Press et al. 1986). Therefore, Gauss-Seidel iteration is always preferred, and the Jacobi iteration is of little value in practice.

Equation (3.56) involves a fully coupled solute transport and geochemical reactions, solved on a node-by-node basis using Gauss-Seidel iteration. This algorithm is hereafter referred to as the *Fully Coupled* algorithm. The size of the Jacobian matrix involved in the *Fully Coupled* algorithm is only $N_s \times N_s$. Even a system with 100 species can be handled easily by most computers. Moreover, the size of the matrix is independent of the number of nodes and, thus, independent of the dimensions of problems. The algorithm is applicable to 1-, 2-, or 3-dimensional domains. Although the *Fully Coupled* algorithm bears the inherent disadvantage of slow convergence from Gauss-Seidel, it does not require large memory that otherwise prevents the problem from solution. Experience shows that combined with a flexible time-step control scheme, the algorithm can solve various practical problems.

3.5.5 Operator Split Algorithm

As an alternative to the *Fully Coupled* algorithm, an *Operator Split* algorithm to solve the reaction-transport equations is available in STORM. As its name suggests, the *Operator Split* algorithm splits the reaction-transport PDE into two parts. One is the transport part that is a linear PDE; the other is the chemical reaction part that is a nonlinear ordinary differential equation (ODE). The algorithm then solves them separately and sequentially (Mangold and Tsang 1991). It has been proved the *Operator Split* algorithm has an inherent mass conservation error (Valocchi and Malmstead 1992). Although a remediation scheme has been proposed by these authors to address that problem, the scheme works only when the boundary condition is homogenous (Barry et al. 1996). Otherwise, it can introduce an even larger error. Another remedial method is to iterate the two steps until they converge (Yeh and Tripathi 1991). This latter approach is implemented in STORM.

In the *Operator Split* algorithm, by splitting the transport part from the reaction part, the solute transport mass conservation equation is decoupled and, thus, can be solved separately. In STORM, when using the *Operator Split* algorithm, the solute transport mass conservation equation is solved as described in Section 3.4. The advantage is that each discretized PDE is a linear equation where the dimension of the coefficient matrix, $\mathbf{b}$, is equal to the number of nodes squared. Therefore, $\mathbf{b}$ is $(N_s)^2$ times smaller than the global matrix $\mathbf{A}$. For the preceding example, it is 9,000,000. Combined with matrix compression techniques, the matrix can be stored in workstations with moderate memory. Because $\mathbf{b}$ can be generated and stored at once, more sophisticated techniques of solving linear equations (such as conjugate gradient or preconditioning conjugate gradient methods) can be used to accelerate convergence.

The second step of *Operator Split* algorithm is to solve the nonlinear ODE at each node to obtain reaction rates for each species. This step is similar to the *Fully Coupled* algorithm. However, unlike the *Fully Coupled* algorithm where the latest updated results of neighboring nodes are used, the *Operator Split* algorithm cannot use them because the latest updated results of neighboring nodes cannot propagate to the current node until the next iteration between the first step and the second step. The solution of the geochemical reaction rates, while using the *Operator Split* algorithm, then effectively uses Jacobi iteration. Therefore, as described in Section 3.5.4, its convergence speed is only half that of the *Fully Coupled* algorithm.

Although the *Operator Split* algorithm is slower than the *Fully Coupled* algorithm, it is well known that contaminant transport simulators based on single-point upstream-weighting for solution of the conservation equations (Section 3.5.3) suffer from a grid size dependent artificial diffusion that smears otherwise sharp fronts. To enable STORM to better manage advection-dominated problems, such as simulating the Pressurized Unsaturated Flow (PUF) experiments, a Total Variation Diminishing (TVD) transport scheme has been implemented in conjunction with the *Operator Split* algorithm.

A flux-limited third-order scheme is applied to the advective-diffusion equations in STORM, following a method that is shown to exhibit close agreement with analytical solutions (Gupta et al. 1991). The difference scheme for the conservation equation is illustrated for 1-dimensional, strictly advective flow with constant grid size. This arrangement results in a strictly hyperbolic conservation equation

$$\frac{\partial c}{\partial t}+u\frac{\partial c}{\partial x}=0 \tag{3.57}$$

where c is the concentration of the aqueous species, t is time, x is distance, and u is the velocity (taken to be a positive constant). Integrating over mesh cell i and time step interval $[n,n+1]$ yields

$$\frac{c_i^{n+1}-c_i^n}{\Delta t}=\frac{u\overline{c}_{i+1/2}-u\overline{c}_{i-1/2}}{\Delta x}=0 \tag{3.58}$$

where $u\overline{c}_{i+1/2}$ represents the time averaged flux through the right edge of the ith mesh cell. The principal problem in the formulation of difference schemes lies in the evaluation of these time-averaged fluxes. The finite-difference approximation may be generalized as

$$\overline{c}_{i+1/2}=c_i^n+\tfrac{1}{2}(1-\lambda)F_i\Delta x \tag{3.59}$$

where λ is the Courant number and F_i is the concentration gradient of the upwind cell. For $F_i = 0$, Equation (3.59) reduces to single-point upstream weighting, the technique used in the *Fully Coupled* algorithm that is only first-order accurate and consequently highly diffusive. A third-order scheme provides higher accuracy (Gupta et al. 1991)

$$F_i=\frac{2-\lambda}{3}\frac{c_{i+1}^n-c_i^n}{\Delta x}+\frac{1+\lambda}{3}\frac{c_i^n-c_{i-1}^n}{\Delta x} \tag{3.60}$$

Although higher-order schemes give better resolution for discontinuities, they exhibit spurious oscillations around these points. The underlying principle of constructing a TVD scheme is to combine lower and higher order fluxes and impose limiting functions on the higher-order flux to prevent formation of local extrema (Harten 1983). The higher order flux is viewed as consisting of the first-order flux, plus a corrective term. Rewrite Equation (3.59) in the following terms

$$\overline{c}_{i+1/2}=c_i^n+\phi(r)\frac{c_{i+1}^n-c_i^n}{2}(1-\lambda) \tag{3.61}$$

where $\phi(r)$ is the flux-limiting function and r is a measure of the smoothness of the data. One possible choice for r is the ratio of consecutive gradients (Sweby 1984). For $u > 0$,

$$r = \frac{c_i^n - c_{i-1}^n}{c_{i+1}^n - c_i^n} \ . \tag{3.62}$$

The third-order scheme can now be obtained by defining $\phi(r)$ appropriately

$$\phi(r) = \frac{2-\lambda}{3} + \frac{1+\lambda}{3} r \tag{3.63}$$

In order to prevent non-physical oscillations, the limiting function $\phi(r)$ must be chosen so that limited anti-diffusive flux is maximized in amplitude subject to the constraint of the resulting scheme being TVD that requires the following equation (Sweby 1984)

$$0 \le \frac{\phi(r)}{r},\ \phi(r) \le 2 \ . \tag{3.64}$$

Equation (3.64) results in the following TVD limiting function for the third-order scheme

$$\phi(r) = \text{MAX}\left[0, \text{MIN}\left(2, 2\text{r}, \phi(r)\right)\right] \tag{3.65}$$

For multidimensional flow problems, the difference scheme given by Equation (3.61) can be applied similarly in the y and z directions.

3.6 Boundary Conditions

The discretization methods described previously for the mass, energy, and solute mass conservation equations strictly applied to interior nodes (that is, nodes surrounded by neighboring nodes). For nodes adjacent to a domain boundary or an inactive node, the discretized forms of the governing equations are modified according to the user specified boundary conditions. Zero flux boundary conditions are applied whenever no boundary condition is specified for a boundary surface. Boundary conditions that can be specified for a boundary surface are varied, depending on the operational mode, and other associated boundary conditions.

A total of eight boundary conditions are appropriate for flow boundaries and are applicable to the conservation equations for water and air: Dirichlet, Neumann, zero flux, initial condition, saturated, unit gradient, hydraulic gradient, and free gradient. The Dirichlet boundary condition specifies the value of intrinsic properties on the boundary surfaces (for example, aqueous pressure and gas pressure). The Neumann boundary condition specifies a surface flux on the boundary surface (for example, aqueous Darcy velocity and gas Darcy velocity). The zero flux boundary condition specifies an impermeable boundary for flow or transport and serves as the default condition for undeclared boundary surfaces (for example, zero fluid flow). The initial condition boundary condition uses the initial conditions in the nodes adjacent to a boundary surface to fix the intrinsic properties on the boundary surface. The saturated boundary condition is appropriate for fluid flow boundaries for 2-phase systems and establishes zero capillary pressure conditions on the boundary surface. The unit gradient boundary condition is appropriate for fluid flow boundaries and establishes a fluid pressure on the boundary surface equal to the fluid pressure at the adjacent node, modified by the hydraulic gradient for the fluid. The free gradient boundary condition is appropriate for fluid flow boundaries for 2-phase systems and establishes a fluid pressure on the boundary surface by extrapolating the local pressure gradient within the computational domain to the boundary surface. The hydraulic gradient boundary condition is appropriate for fluid flow boundaries and establishes a series of boundary pressures according to the local hydraulic gradient for the fluid.

The five boundary conditions appropriate for transport boundaries and applicable to the conservation equations for energy and solute mass follow: Dirichlet, zero flux, initial condition, outflow, and inflow. The Dirichlet boundary condition specifies the value of intrinsic properties on the boundary surfaces (for example, temperature or solute concentration). The zero flux boundary condition specifies an impermeable boundary for flow or transport and serves as the default condition for undeclared boundary surfaces (e.g., zero heat flux, or zero solute flux). The initial condition boundary condition uses the initial conditions in the nodes adjacent to a boundary surface to fix the intrinsic properties on the boundary surface. The outflow boundary condition considers transport out of the computational domain by advection only, so no diffusive transport occurs. This boundary condition will not transport energy or solute mass into the domain. The inflow boundary condition considers transport into the computational domain only by advection and no diffusive transport occurs. This boundary condition will not transport energy or solute mass out of the domain.

In general, boundary conditions for simulations that involve the solution of multiple governing equations can be combined in a variety of ways, with one major restriction. Specification of a Neumann boundary condition for the energy equation on a boundary surface (other than the zero flux condition) requires that flow equation boundary conditions for that boundary surface be specified as zero flux. Nonzero fluid flow and heat transport can be achieved for boundary nodes through a combination of flow boundary conditions, a zero flux energy boundary, and energy sources for the boundary nodes. Regardless of the boundary condition or combination of boundary conditions a sufficient number of independent intensive variables must be declared to specify the thermodynamic and hydrologic state on the boundary surface. Definitions for geometric parameters for nodes with boundary surfaces on the west side are shown in Figure 3.3 for the X-Z Cartesian coordinate plane and in Figure 3.4 for the X-Y Cartesian coordinate plane.

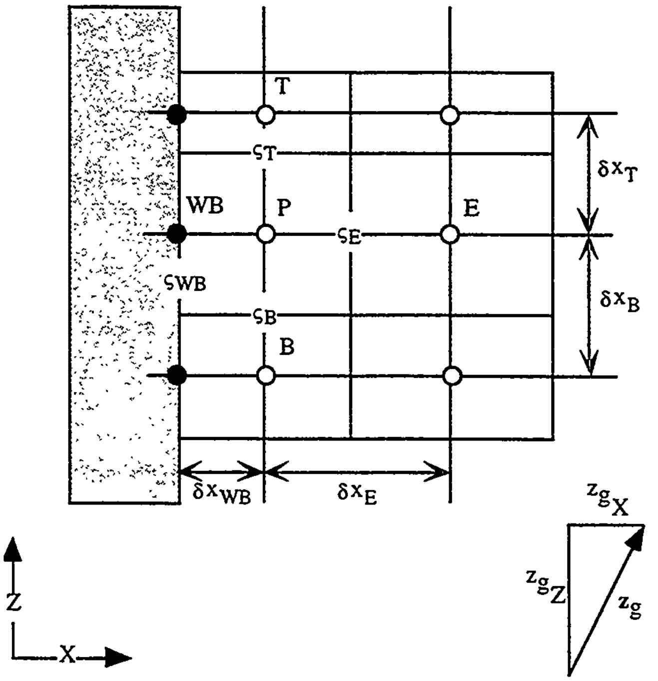

Figure 3.3. X-Z Cartesian Coordinate Plane for West Boundary

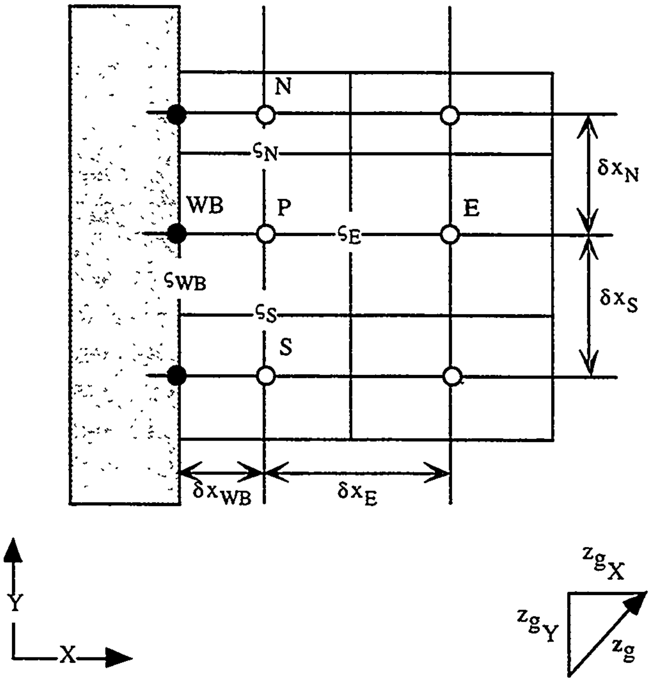

Figure 3.4. X-Y Cartesian Coordinate Plane for West Boundary

3.6.1 Dirichlet Boundary Condition

The Dirichlet boundary condition is equivalent to specifying the value for the primary unknown on the boundary surface. Assigned values of primary variables are used to compute secondary variables for the boundary surface. Average properties for transport between a boundary surface and the adjacent node are computed with user-specified averaging schemes between the values of the properties on the boundary surface and adjacent node. For example, the discretized forms for Darcy mass flux and diffusive mass flux rates of water, Equation (3.6), are modified for a Dirichlet aqueous phase boundary condition on a west boundary surface, according to Equations (3.66) and (3.67), respectively. For the solute mass conservation equation, a Dirichlet boundary condition on the west surface produces a modification to the discretized conservation equation, Equation (3.26), according to Equation (3.68).

$$F_{l\,WB}^{w} = -\frac{\left\langle \omega_l^{w} \rho_l k_{rl} \right\rangle_{WB}^{uw} \left\langle \mathbf{k} \right\rangle_{WB}^{h}}{\left\langle \mu_l \right\rangle_{WB}^{h}} \left(\frac{\left(P_{lP} - P_{lWB}\right)}{\delta x_{WB}} + \left\langle \rho_l g \right\rangle_{WB}^{a} z_{gWB} \right) \tag{3.66}$$

$$J_{l\,WB}^{w} = -\left\langle \tau_l n_D \rho_l s_l \frac{M^{w}}{M_l} \mathbf{D}_l^{w} \right\rangle_{WB}^{h} \frac{\left(\chi_{l\,P}^{w} - \chi_{l\,WB}^{w}\right)}{\delta x_{WB}} \tag{3.67}$$

$$a_P C_P - \sum_{\varsigma=E,S,N,B,T} a_\varsigma C_\varsigma = \frac{\partial C}{\partial t}\overline{V} - \dot{m}^{C}\overline{V} - \dot{R}^{C} C\overline{V} + a_{WB} C_{WB} \tag{3.68}$$

3.6.2 Neumann Boundary Condition

The Neumann boundary condition is equivalent to specifying the flux on a boundary surface. Fluxes that may be specified are aqueous volumetric, gas volumetric, energy, and solute mass. For example, the discretized form for Darcy mass flux of water, Equation (3.6), is modified for a Neumann boundary condition on a west boundary surface, according to Equation (3.69). Sufficient information is needed to fix the thermodynamic and hydrologic state on the boundary surface. Calculation of phase pressures from phase volumetric flow rates requires an iterative solution because averaged values of properties (for example, relative permeability) are nonlinear functions of the phase pressure on the boundary. To avoid an iterative solution of the phase pressure on the boundary phase, pressures are computed assuming a unit relative permeability, according to Equation (3.70) for a west boundary surface.

$$F_{l\,WB}^{w} = \left\langle \omega_l^{w} \rho_l \right\rangle_{WB}^{uw} V_{lWB} \tag{3.69}$$

$$P_{lWB} = \delta x_{WB} \left(\frac{V_l^{w}{}_{WB} \langle \mu_l \rangle_{WB}^{h}}{\langle \mathbf{k} \rangle_{WB}^{h}} + \langle \rho_l g \rangle_{WB}^{a} \, z_{gWB} \right) + P_{lP} \tag{3.70}$$

3.6.3 Zero Flux Boundary Condition

The zero flux boundary condition is the default boundary condition and is equivalent to specifying zero flow or transport across the boundary surface. Complex boundary conditions that involve multiple specifications on a single boundary surface, require that sufficient information be specified to fix the thermodynamic and hydrologic state on the boundary surface. For example, a recognized boundary condition combination for a water-air system (that is, the Two-Phase Volatile operational mode) is a Dirichlet aqueous boundary and zero flux gas boundary. Isothermal water-air systems require two independent intensive variables to fix the thermodynamic and hydrologic state of the system. For this boundary system, the Dirichlet aqueous boundary assigns a value to the aqueous pressure. The gas pressure can be specified by user input or be computed using zero flux boundary conditions and solving Equation (3.6) for the gas pressure, according to Equation (3.71) for a west boundary surface.

$$P_{gWB} = \delta x_{WB} \langle \rho_g g \rangle_{WB}^{a} \, z_{gWB} + P_{gP} \tag{3.71}$$

3.6.4 Initial Condition Boundary Condition

The initial condition boundary condition is identical in application to the Dirichlet boundary condition, with the following exception: the primary variable is fixed to the initial value at the adjacent node. No user input is required for this type of boundary condition because boundary values are obtained through the initial condition specifications. Initial boundary pressures are computed once at the start of a simulation.

3.6.5 Saturated Boundary Condition

The saturated boundary condition is recognized for water-air systems and performs as do dynamic Dirichlet boundary conditions, where zero capillary pressure is maintained on the boundary surface. The saturated boundary condition fixes the aqueous pressure equal to the gas pressure on a boundary surface, regardless of the boundary condition for the gas pressure. For a single-phase system, the gas pressure is fixed through the initial conditions and the aqueous pressure is maintained on a saturated boundary equal to this gas pressure. For a 2-phase system, the gas pressure on a boundary surface is user-specified according to the gas-phase boundary conditions. The saturated boundary condition for the aqueous phase fixes the aqueous pressure at the boundary surface equal to the gas pressure. Saturated boundary pressures are computed with each Newton-Raphson iteration.

3.6.6 Unit Gradient Boundary Condition

The unit gradient boundary condition is recognized for aqueous-phase and gas-phase boundary conditions and maintains a unit gradient in the phase hydraulic head. A unit gradient in the phase hydraulic head is equivalent to setting the normalized Darcy velocity equal to minus one, according to Equation (3.72) for a west boundary surface. Phase pressure on the boundary surface is computed by solving Equation (3.72) for the boundary pressure, according to Equation (3.73), for a west boundary surface. Unit gradient boundary pressures are computed with each Newton-Raphson iteration.

$$\frac{V_{\gamma WB} \langle \mu_\gamma \rangle^{h}_{WB}}{\langle k_{r\gamma} \rangle^{uw}_{WB} \langle \mathbf{k} \rangle^{h}_{WB}} = -1 = -\left(\frac{\left(P_{\gamma P} - P_{\gamma WB}\right)}{\delta x_{WB}} + \langle \rho_\gamma g \rangle^{a}_{WB} z_{gWB} \right) \text{ for } \gamma = l, g \tag{3.72}$$

$$P_{\gamma WB} = P_{\gamma P} + \delta x_{WB} \left(\langle \rho_\gamma g \rangle^{a}_{WB} z_{gWB} \right) \text{ for } \gamma = l, g \tag{3.73}$$

3.6.7 Free Gradient Boundary Condition

The free gradient boundary condition is recognized for aqueous-phase and gas-phase boundary conditions. This boundary condition is essentially a dynamic Dirichlet-type boundary, where the pressure on the boundary surface is set to maintain the pressure gradient in the interior nodes adjacent to the boundary surface. This boundary type requires, as a minimum, two active nodes adjacent to the boundary surface. The gradient in phase pressure is linearly extrapolated from the interior nodes to the boundary surface to determine the boundary pressure. Free gradient boundary pressures are computed with each Newton-Raphson iteration.

3.6.8 Outflow Boundary Condition

The outflow boundary condition is recognized for transport boundaries (for example, energy transport, solute transport). This boundary condition allows transported quantities to be moved across a boundary surface by advection only. Inward transport by diffusion or transport across the boundary surface is prohibited. Energy flux across an outflow boundary surface on a west boundary is computed according to Equation (3.74). Solute flux across an outflow boundary surface on a west boundary is computed according to Equation (3.75).

$$Q_{\gamma WB} = -\left(\rho_\gamma h_\gamma\right)_P \max\left[-V_{\gamma WB}, 0\right] \text{ for } \gamma = l, g \tag{3.74}$$

$$G^{C}_{\gamma\, WB} = -C_{\gamma P} \max\left[-V_{\gamma WB}, 0\right] \text{ for } \gamma = l, g \tag{3.75}$$

3.6.9 Inflow Boundary Condition

The inflow boundary condition is recognized for transport boundaries (for example, energy transport and solute transport). This boundary condition allows transported quantities to be inwardly transported across a boundary surface by advection only. Transport by diffusion or transport outward across the boundary surface is prohibited. Energy flux across an inflow boundary surface on a west boundary is computed according to Equation (3.76). Solute flux across an inflow boundary surface on a west boundary is computed according to Equation (3.77).

$$Q_{\gamma WB} = \left(\rho_\gamma h_\gamma \right)_P \max\left[V_{\gamma WB}, 0 \right] \text{ for } \gamma = l, g \tag{3.76}$$

$$G^C_{\gamma\, WB} = C_{\gamma P} \max\left[V_{\gamma WB}, 0 \right] \text{ for } \gamma = l, g \tag{3.77}$$

4.0 Input File

4.1 Introduction

The STORM simulator is controlled through a text file that must be named *input* for proper execution. This input file has a structured format composed of cards that contain associated groups of input data. Depending on the operational mode, input cards might be required, either 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 simulation. Unused cards are treated as additional text that is not recognized by the simulator, but does 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 Appendix A. Data structures within cards vary with the operational mode, which requires the user to follow a series of logic type statements to construct a readable input file. The STORM simulator contains logic to capture and report input errors. These capabilities are primarily limited to indicating syntax or formatting type errors and generally do not reveal such errors as those associated with ill-posed problems, atypical parameters, or inappropriate grid structures, for example. Because of its text format, STORM input files are generated with text editors, word processors, spread sheet programs, or graphical interfaces and are portable between computing environments. The input is case insensitive and allows 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 STORM input file is composed of cards, some of which are required and others that are optional or unused. The number of required cards depends on the operational mode. If an attempt is made to execute the STORM 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 the STORM 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 STORM simulator on input files with an incomplete set of optional cards yields messages that note the missing optional cards and allow the execution to continue. A summary of the required and optional cards that compose a STORM input file for each operational mode is shown in Table 4.1.

Each card is delimited on the first line with a card header that must contain a tilde symbol (for example, ~ Simulation Title Card) as the character in the first column of the line. Cards may be arranged in any order within an input file; however, the input format within a card is generally structured. If a card type is repeated within an input file, the STORM simulator reads only the first card. Blank lines or additional comment lines can be included in the input file outside of the card structures.

Table 4.1. Required and Optional Input Cards

Operational Mode	Card Name	Card Status
All	Initial Conditions	Optional
	Boundary Conditions	Optional
	Source	Optional
	Output Control	Optional
	Surface Flux	Optional
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
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
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

4.3 Formatting and Notation

Each input card has formatting specifications that must be followed to create an input file readable by the STORM simulator. The format structure of certain cards varies with the operational mode or other input options, and the user must strictly follow the formatting instructions for the particular operational mode of interest. As expected, the complexity of the input increases with the operational mode. Guides for formatting input cards and card lines are presented in Appendix A. The formatting guides shown in this appendix were written in a compact format using special notations. Definitions of the formatting notations used in Appendix A 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 commas delimit data. The STORM simulator recognizes three different types of data, integers, real data, and character strings. Integers are used primarily to indicate indexes 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 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 Appendix A using a format guide (for example, **Format:** *Integer*^{a}, *Real*^{b}, *Char*^{c},). Data types in the format statements are indicated as *Integer*, *Real* and *Char* (that is, integer, real, and character string) with lowercase letter superscripts. The lettered superscripts are used to correlate the data item with its position in the input line. The total length of a single input line cannot exceed 132 characters. Additional text may appear after the closing comma of an input line. Many input variables contain default values (indicated with underlining in Appendix A). Using a *null* entry accesses a default value. The *null* entry requires the closing comma for the input data item to immediately follow the previous closing comma or that 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 Appendix A after the formatting instructions for each card. A considerable portion of the input file formatting and creation work is eliminated by developing input files from previously generated files or through an input generator.

4.4 Units

The STORM simulator offers considerable flexibility in specifying units. The user can declare units for both input or output data. Unspecified units are assumed to be in standard Systeme Internationale (SI) units for the data item. The STORM simulator operates internally almost exclusively in SI units, with pressures expressed in gauge, relative to 1 atmosphere. Unless specifically stated, all input and output pressures are expressed in absolute values. Unit variables are read by the STORM 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 or a single divisor symbol (that is, /). 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 part of the numerator and, conversely, the unit strings following the divisor symbol are considered part of the denominator. Units recognized by the STORM simulator are listed in Table 4.3.

Table 4.2. Input Format Notation Guide

Notation	Description
{ Option }	Enclosing braces indicate character string options. 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.
{{ Fractured }}	Indicates the word fractured is within the rock/soil type name, thus indicating a dual porosity model.
< 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 that is 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.
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.

Table 4.2. Input Format Notation Guide (contd)

Notation	Description
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	$1.\times10^5$	$M / L T^2$
btu	Btu	J	1054.4	$M L^2 / T^2$
c	Celsius	C	1.	K
cal	calorie	J	4.184	$M L^2 / T^2$
ci	Curie	N/A	N/A	N/A
cm	centimeter	m	$1.\times10^{-2}$	M
cp	centipoise	Pa s	$1.\times10^{-3}$	$M / L T$
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
degree	degrees	rad	0.017453	N/A
dyn	dynes	N	$1.\times10^{-5}$	$M L / T^2$
dynes	dynes	N	$1.\times10^{-5}$	$M L / T^2$
f	Fahrenheit	C	(f-32)×(5/9)	K
ft	foot	m	0.3408	L
furlong	furlong	m	2.01168×10^2	L
g	gm	kg	$1.\times10^{-3}$	M
gal	gallon	m^3	3.7854×10^{-3}	M^3
gas	gas phase	N/A	N/A	N/A
gm	gm	kg	$1.\times10^{-3}$	M

Table 4.3. Recognized Units (contd)

Notation	Description	SI Equivalent	Conversion to SI	Base Units
gram	kg	kg	$1.\times10^{-3}$	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	m^3	$1.\times10^{-3}$	L^3
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	$1.\times10^{-3}$	L^3
m	meter	m	1.	L
mg	milligram	kg	$1.\times10^{-6}$	M
min	minute	s	1/60.	T
ml	milliliter	m^3	$1.\times10^{-6}$	L^3
mm	millimeter	m	$1.\times10^{-3}$	L
mol	mole	kgmol	$1.\times10^{-3}$	Mo
mole	mole	kgmol	$1.\times10^{-3}$	Mo
p	Poise	Pa s	$1.\times10^{-1}$	M / L T
pa	Pascal	Pa	1.	$M / L T^2$
pci	pico Curies	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	N/A	N/A	N/A
solid	solid	N/A	N/A	N/A
w	Watt	W	1.	$M L^2 / T^3$
water	water	N/A	N/A	N/A

Table 4.3. Recognized Units (contd)

Notation	Description	SI Equivalent	Conversion to SI	Base Units
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

The units listed under the subtitle *Miscellaneous Units* are primarily descriptive units that may be included in a unit character string to increase its readability. The *hc* unit, however, is uniquely reserved for specifying rock/soil hydraulic conductivity that is normally expressed in velocity units. Without the *hc* unit, the input for rock/soil permeability is interpreted as an intrinsic permeability value. The STORM 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 conductivity, the associated unit character string should include the *hc* unit (for example, hc cm/hr).

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 enters the character string Btu in/hr ft^2 F for the unit data item. Note the carat (^) symbol is used to indicate an exponential. During simulator execution, the string of English units for thermal conductivity is translated into the primary unit form, m/s^3 kg K, and compared against the standard unit form. If the primary unit form does not agree with the standard unit form, an error message is generated by the STORM simulator and the program execution stops. Otherwise, if the comparison is successful, the input data for thermal conductivity, that precedes its unit string, is converted to SI units.

4.5 Card Descriptions

Formatting instructions for the input cards are provided in Appendix A. 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 Appendix A.

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 that 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 record becomes especially valuable, when the user is making repeated simulations with small modifications to the input parameters. The changes required 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 file write sequence, and have name extensions that correspond to the generating time step (for example, the file *restart*.28 is generated after 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/Transport* 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* 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 that are solved by the simulator. Because execution performance is indirectly proportional to the number of solved governing equations, the user selects an *Operation Mode* most appropriate for the problem of interest. For example, if the physical system to be modeled does not contain thermal or gas pressure gradients, the most appropriate operational mode is the Water operational mode, where only the water mass conservation equation is solved. Executing the described physical system under the *Water-Air* operational mode yields nearly identical results; however, the execution time is significantly increased. Considerable attention is 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 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, indicated by including the keyword Transport in the operational mode (for example, *Water-Air* and *Solute Transport*). Solute transport is solved using the Patankar method, unless the keyword TVD appears, indicating the TVD method.

Execution periods refer to a period of simulation time. The STORM simulator allows the user to specify single or multiple execution periods. For each execution period, the user can control the initial time step, maximum time step, time step acceleration factor, maximum number of Newton-Raphson iterations and convergence criterion. Recommended value for the *Time Step Acceleration Factor* is 1.25, the *Maximum Number of Newton-Raphson Iterations* is 8, and the *Convergence Criterion* is 1.e-06. 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 advisable to change the value for the *Convergence Criterion* from its recommended value. This value has proven through numerous applications to achieve a preferred balance between accuracy and execution speed.

Field variables, including 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. Such values 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 STORM simulator are shown in Table 4.4.

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
Gas density	Upwind
Gas relative permeability	Upwind
Gas viscosity	Harmonic
Hydraulic dispersion	Harmonic
Intrinsic permeability	Harmonic
Solute diffusion	Harmonic
Thermal conductivity	Harmonic
Water vapor diffusion	Harmonic

For simulations of physical systems involving heat transfer, note 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 finite-difference formulation of the STORM simulator 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 can 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 and the radial (r–direction), and azimuthal (Q-direction) axes 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-, Q-, and z-directions, respectively, and the terms East, North, and Top refer to the positive r-, Q-, and z-directions, respectively. Negative dimensional values are not recognized and axes are defined as positive toward increasing node numbers. The grid dimensions that are specified on the Grid Card refer to node surfaces; therefore, for grids with non-uniform 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 that are less than 360 degrees. Refer to the STOMP Theory Manual (White and Oostrom 1996) for graphical descriptions of the Cartesian and cylindrical coordinate systems.

4.5.4 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 that must contain no more than 64 characters. Because most physical systems of interest contain formations or zones of rocks or soils, the computational domain also contains zones of nodes with similar hydrogeologic properties. With this card, the user defines the distribution of rock/soil zones. In subsequent input cards, hydrogeologic properties are defined with respect to the defined rock/soil types. This approach avoids having to specify hydrogeologic properties uniquely for every node. Generally, this approach is preferred. 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. 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, *Gas 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 initially declares 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 often avoids undeclared rock/soil types.

4.5.5 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/m^3 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 only to 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 with a default value of $1.\text{x}10^{-7}$ 1/Pa for the compressibility. Tortuosity functions are required for simulations that involve solute transport or diffusion of components through phases (for example, water vapor diffusing through the gas phase). Tortuosities can be computed as either constants that require input values, or as functions of the phase saturation and diffusive porosity through the Millington and Quirk function. Refer to the STOMP Theory Manual (White and Oostrom 1996) for a description of the Millington and Quirk tortuosity function.

4.5.6 Hydraulic Properties Card

This card allows the user to assign values to the intrinsic permeability for 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 STORM 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. With the STORM simulator, a primary assumption 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 Manual (White and Oostrom 1996) for a description of the conversion of hydraulic conductivity to intrinsic permeability at reference conditions.

4.5.7 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 Manual (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 STORM 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.

4.5.8 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. Although functional forms for the saturation-capillary pressure functions are preferred, tabular input is acceptable. By default, tabular data is interpolated using linear interpolation. Values beyond the table limits will be assigned either the table minimum or maximum values, as appropriate. 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 from the relative permeability function cards for the liquid phases. Hysteretic functions are those for which the drainage and imbibition scanning paths differ and that include capabilities for entrapment of fluids of lower wettabilities. Non-hysteretic functions require fewer input items than their hysteretic counterparts and use a single scanning path for drainage and imbibition events. Refer to the STOMP Theory Manual (White and Oostrom 1996) for a complete description of saturation-capillary pressure functions.

4.5.9 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 λ *Parameter* can be defaulted to the values entered or defaulted with the saturation function. Although functional forms for the relative permeability-saturation functions are preferred, tabular input is acceptable. By default, tabular data is interpolated using linear interpolation. Values beyond the table limits are assigned appropriately either the table minimum or maximum values. Refer to the STOMP Theory Manual (White and Oostrom 1996) for a complete description of aqueous relative permeability-saturation functions.

4.5.10 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 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 are interpolated using linear interpolation; cubic spline interpolation is optional. Values beyond the table limits are assigned either the table minimum or maximum values, as appropriate. Refer to the STOMP Theory Manual (White and Oostrom 1996) for a complete description of gas relative permeability-saturation functions.

4.5.11 Solute/Fluid Interaction Card

This card allows the user to define solutes, solute-fluid interactions, and solute radioactive decay paths. It is required only for simulations involving solute transport. The STORM 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 the definition of their progeny. The STORM simulator actually solves the transport equation for each solute sequentially and in reverse order from the definition list on this card. Solutes are defined by a unique solute name that 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 that produces a particular progeny. *Chain Decay Fraction* inputs are fractional values from 0.0 through 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 STORM simulator. The interphase partition coefficients define the equilibrium distribution of solute between the active liquid phases. Partition coefficients that define the equilibrium adsorption of solute onto the solid phase are determined on the *Solute/Porous Media Interaction* Card. It is important for the user to carefully note the definitions and requested units for each interphase partition coefficient. The *Aqueous-Gas Partition Coefficient* is the ratio of the solute mass in the aqueous phase per unit mass of aqueous phase to the solute mass in the gas phase per unit volume of gas phase. Refer to the STOMP Theory Manual (White and Oostrom 1996) for a more detailed description of the interphase partition coefficients, solute diffusion coefficients, radioactive decay rate equations, and radioactive chain decay fractions.

4.5.12 Solute/Porous Media Interaction Card

This card allows the user to define solid-aqueous phase partition coefficients and porous media dependent hydraulic dispersivities. It 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 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. This definition refers to the concentration of solute adsorbed on the solid phase per unit mass of solid phase with 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 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 Manual (White and Oostrom 1996) for a more complete description of the solid-aqueous interphase partitioning and hydraulic dispersion of transported solutes.

4.5.13 Initial Conditions Card

This card allows the user to assign starting values to primary and secondary field variables by initializing primary variables. However, the current version of the STORM simulator does not allow the user to initialize flux or surface variables. Such initializations are useful for solving transport problems with steady flow fields. Transport problem simulations with the STORM simulator require the solution of the flow fields. The variables that may be initialized are dependent on the operational mode. For all operational modes, default values for the initial conditions are specified within STORM as shown in Table 4.5.

For the *Water*, *Water-Air*, and *Water-Air-Energy* operational modes, STORM 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 is computed from the other two through the declared saturation-capillary pressure functions. Initial conditions are

Table 4.5. Initial Condition Default Values

Initial Field Variable	Symbol	Default Value
Temperature (C)	T	20.
Pressure of phase j (Pa)	P_j	101325.
Saturation of phase j	s_j	0.
Mole fraction of component i in phase j	χ_j	0.
Solute concentration in phase j	Sj	0.

declared repeatedly for a node with the application of last definition. 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*), or the gas-phase volume (*Gas Volumetric*). Solute units are undefined and expressed as the user chooses (for example, Ci, pCi, gm, kg, mol, or kgmol). Although units for expressing solute quantity 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 are used to advantage whenever possible. Gradients to the initial conditions allow the user to specify that initial value of a field variable 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 indexes. If nonzero gradient values are specified, the initial condition values 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 initial condition domain of the state. An exemplary application of the initial condition gradient utility occurs for problems that start with hydrostatic conditions. For these problems, the pressure at the lowest z-direction node positions is entered with a *Z-Dir. Gradient* that equaled minus the fluid density times the acceleration of gravity. For water at 20°C, a z–direction gradient of -9793.5192 1/m yields constant head conditions, that is the equivalent to hydrostatic conditions. This approach is used to locate the position of a water table under no-recharge equilibrium conditions, given the rock/soil saturation function properties and a single pressure using an *Initial Condition* execution mode simulation.

4.5.14 Boundary Conditions Card

This optional card allows the user to control the simulation by defining time varying boundary conditions. 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 flow and transport. Boundary conditions can 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 location and a direction with respect that 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 vary within periods. The user is not allowed to assign multiple boundary conditions to a boundary surface during the same time period, but multiple boundary conditions do so over different time periods. The STORM 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 STORM simulator allow the user the flexibility to solve most subsurface flow and transport problems. In addition, the boundary condition card reader within the STORM simulator performs limited error checking on the boundary condition inputs. However, an error-free boundary condition card does not guarantee the user has not created either an ill-posed problem or an execution that successfully converges. For example, a mistake frequently made by STORM users is to specify infiltration rates at the top of a column with positive fluxes. Although this input is perfectly acceptable to the STORM boundary condition input reader, the specified condition actually withdraws flux from the top of the column because 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-, Q-, and z-directions, respectively, and the terms East, North, and Top refer to the positive r-, Q-, 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 types that 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 (for example, pressure, temperature, or solute concentration) at the boundary surface. The *Neumann* boundary type allows the user to specify a flux (for example, liquid phase flux, heat flux, or solute flux) at the boundary surface. The *Zero Flux* boundary type is used to impose no flow or transport conditions across the boundary. The *Saturated* boundary type is available only for 2-phase conditions and imposes total-liquid saturation conditions (for example, water table) at the boundary surface. The *Unit Gradient* boundary type imposes hydrostatic conditions across the boundary surface for the specified phase. The *Free Gradient* boundary type is available only for the *Water* operational mode and models a free drainage surface with respect to the aqueous phase. The *Hydraulic Gradient* boundary type is 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. The simulator then computes fluid phase pressure for the remaining boundary surfaces assuming hydrostatic conditions for the fluid phase. The *Initial Conditions* boundary type fixes the boundary field variables (for example, pressure, temperature, or solute concentration) to the initial values of the field variables at the node adjacent to the boundary surface. This boundary type is invariant with time. *Outflow* boundary types are applicable only to solute and energy boundary conditions. This boundary type considers only advectively transported solute or energy to exit the computational domain. Diffusion transport across the boundary surface is neglected. The solute concentration boundary types (for example, *Volumetric Concentration*, *Aqueous Conc.*, or *Gas 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 the boundary condition is time invariant; the specified *Boundary Time* represents the start time for the boundary condition. Prior to the start time, the boundary surface type is assumed to be *Zero Flux.* The specified boundary condition remains in effect from the start time until the execution completion. If a boundary condition is declared with multiple *Boundary Times*, 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 are simulated by defining duplicate *Boundary Times*; the first time indicates the completion of the previous boundary condition and the second time indicates the start of the new boundary condition. At the completion of the step boundary condition, another set of duplicate *Boundary Time* declarations is used. Step boundary conditions are convenient methods for introducing slugs of fluids, heat, or solute in conjunction with the *Neumann* boundary type.

4.5.15 Source Card

This card allows the user to control sources or sinks of mass, energy, and solutes by defining the 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, application of multiple sources to a node are allowed during the same time period. The STORM 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 an 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. For non-isothermal problems, the injected fluid also transports an amount of heat into the node, based on the enthalpy of the entering aqueous fluid. Sources of types *Gas Volumetric/Mass Fraction*, *Gas Volumetric/Relative Humidity*, *Gas Mass/Mass Fraction*, and *Gas Mass/Relative Humidity* are similar to the two aqueous sources, in that mass and heat injected into the node include contributions from water vapor and air. The source type suffix *Mass Fraction* indicates that water vapor and air concentrations in the gas phase are specified through inputs of mass fractions for these quantities. Similarly, the source type suffix *Relative Humidity* indicates that water vapor and air vapor concentrations in the gas phase are specified through inputs of relative humidities for the water. Energy sources are specified as either 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 (for example, Ci, pCi, gm, kg, mol, or kgmol).

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 are obtained from a restart file. A source declared with a single *Source Time* implies the source is time invariant; the specified *Source Time* represents the start time for the source. Prior to the start time, the source is zero and from the start time to execution completion, the source is as specified. If a source is declared with multiple *Source Times*, 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 are simulated by defining duplicate *Source Times*; the first time indicates the completion of the previous source and the second time indicates the start of the new source. At the completion of the step source another set of duplicate *Source Time* declarations is used. Step sources are convenient methods for introduction slugs of fluids, heat, or solute into an interior node.

4.5.16 Output Control Card

This card allows the user to control output written to the *output* file, *plot* file, and *screen* (that is, 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* 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* file always is generated at the conclusion of an execution. The output to the STDIO primarily comprises variable results of the reference node versus the simulation time or time step. The user is advised to request *screen* output, because well-chosen output may be invaluable in tracking the simulation progress and identifying possible input errors. If a suite of repetitive simulations is in progress, 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. However, 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 indexes that indicate the x-, y-, and z-direction coordinates of the node. Node numbering in the STORM simulator increments in the order i, j, and k, where the indexes refer to the x-, y-, and z-directions, respectively, for Cartesian coordinate systems and r-, q-, and z-directions, respectively, for cylindrical coordinate systems. *Reference Node Screen Output Frequency* and *Reference Node Output File Frequency* are parameters that indicate how often, with respect to time step, reference node output is written to STDIO and the *output* file. 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 controls 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 are expressed in units of seconds and meters, respectively. These inputs allow the user to customize time- and length-scale units to those most appropriate for 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 that appear in field and flux variable results written to the STDIO, *output* file, and *plot* file. Default values for the number of significant digits is 5; the minimum number of significant digits is 4. Field and flux variables output for the reference nodes are selected from the list shown in Appendix A for each operational mode. The same list of variables for each operational is available for output to the *plot* files. Output units for all variables that have units can be specified immediately following the variable name. Variables without units require a *null* entry for the variable units. The *null* entries for variables with units yield default output units that are expressed in SI units.

Plot files are written by default at the conclusion of an execution and at each requested *Plot File Output Time*. Plot files contain geometrical 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* file is generated with every *plot* file. Both restart and plot files are suffixed with an file name extension of a dot followed by an integer (for example, plot.567, restart.32). The extension integer corresponds with the time step for which the file was written. Both *plot* and *restart* files are written at the conclusion of a time step. Field and flux variables recorded to *plot* files are selected from the list shown for each operational mode in Appendix A (refer to the reference node variable options). Output units for all variables with units are 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 that are expressed in SI.

4.5.17 Surface Flux Card

This card allows the user to define surfaces to track fluxes of fluid mass, fluid volume, heat, 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 that 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; 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-, Q-, and z-directions, respectively, and the terms East, North, and Top refer to the positive r-, Q-, and z-directions, respectively. For example, a surface to track the flux rate and integral of a particular solute entering the water table is 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.

4.5.18 Reactive Transport Control Card

This card lists several parameters that control the execution of the reactive transport subroutines. The first line requires specification of the reactive transport *Execution Mode*. Currently, only one *Execution Mode* is available for the reactive transport subroutines: Normal. In the *Normal* mode, the simulation executes from initial conditions specified with the *Reaction IC Card.*

The second line requires the user to select the reactive transport *Solution Mode*. Two solution modes are available: *Fully Coupled mode*, in which the transport equations and the chemical reactions are solved simultaneously, and *Operator Split* mode, in which advective-diffusive transport equations and the chemical reactions are solved sequentially.

The third line lists eight integer values that must be set to either zero or one (Table 4.6).

Table 4.6. Description of Reactive Transport Execution Control Switches

Description Of Variable	0	1
Aitken extrapolation scheme	off	on
Column scan switch for 2-D problems	off	on
Non-reacting solver switch	off	on
Effective reaction area	off	on
Texture solver coupled with transport solver	off	on
Porosity updated with precipitation/dissolution	off	on
Lower boundary condition for all species	No flux	C = 0
Water consumption in chemical reactions coupled with water flow	off	on
Gas consumption in chemical reactions coupled with gas flow	off	on

The "Aitken extrapolation scheme" accelerates the convergence of the Newton-Raphson iteration scheme related to the reactive transport solver. The "Column scan switch for 2-D problems" enables the Gauss-Seidel iterative solver to scan in the horizontal, as well as the vertical direction when solving 2-dimensional problems. The "non-reacting solver switch" allows the user to solve problems with pure transport (no reactions) in the *Fully Coupled* solution mode. The "Effective reaction area" switch allows recalculation of the effective reaction surface area for each mineral at each node, as the minerals precipitate or dissolve. The "Texture solver coupled with transport solver" switch allows the recalculation of the mineral volume and radius as minerals precipitate and dissolve in slow reactions. The "Porosity updated with precipitation/dissolution" switch allows changes in porosity to be passed back to the water or gas flow subroutines. The "Lower boundary condition for all species" switch allows either a no flux or a specified concentration of zero at the lower boundary for all aqueous species. The "Water consumption in chemical reactions coupled with water flow" switch allows the net water consumption rate, due to all chemical reactions, to be coupled with the water flow subroutines. The "Gas consumption in chemical reactions coupled with water flow" switch allows the net gas consumption rate, due to all chemical reactions, to be coupled with the gas flow subroutines.

The fourth line lists seven real variables that control the convergence of the reactive transport solver.

The Courant number is defined as $Cr = V\Delta t/\Delta l$, where Δt is the initial time step, Δl is the minimum horizontal and vertical grid spacing, and V is the average linear pore water velocity. The time stepping parameter is used to select the time-stepping scheme for reactive transport: a value of 0.0 gives an explicit scheme, 0.5 gives a Crank-Nicholson scheme, and 1.0 provides a fully implicit scheme. A fully implicit scheme is likely to be the most stable. The other six input variables allow the user to specify the maximum error allowed in the solution of various reactive transport subroutines.

4.5.19 Aqueous Species Card

This card lists the aqueous species to be considered in the simulation. Examples of aqueous species include ions, such as HCO_3^- and dissolved gases, such as $O_2(aq)$. Required input includes the overall

Table 4.7. Reactive Transport Convergence Control Variables

Description Of Variable	Range Of Appropriate Values
Initial time step Courant number	1 to 1.e-6
Time stepping parameter	0.0 to 1.0
Error in non-reacting species	1.e-2 to 1.e-7
Error in equation point solver	1.e-2 to 1.e-7
Error in transport point solver	1.e-2 to 1.e-7
Error in texture solver	1.e-2 to 1.e-7

charge of the aqueous species and the hard core diameter. Values of these input parameters for many aqueous species are found in the EQ3/6 database file data0.com.R2, for example. Also required is a list of the elements contained in each species.

4.5.20 Gas Species Card

This card lists gas phase species to be considered in the simulation. Examples of gas phase species include $CO_2(g)$ and $O_2(g)$. These species must be related by Henry's Law Constant to an aqueous species, specified in the Aqueous Species Card. Required input includes the diffusion coefficient of the gas phase species and the Henry's Law equilibrium coefficient. The Henry's Law equilibrium coefficient may be specified as a constant or as a polynomial function of temperature, with a user-specified number of terms. Additionally, a list of the elements contained in each gaseous species is required.

4.5.21 Components Card

This card lists the components that consist of the groups aqueous species related by a single element. Components provide a way to keep track of the total mass of a given element in solution. For instance, the component Al(total) can consist of two aqueous species: $Al(OH)_3(aq)$ and $Al(OH)_4^-$.

4.5.22 Solid Species Card

This card lists and describes all solid species in the simulation. Examples of solid species include silicate or carbonate minerals, metals or soil organic matter. Solid species are defined by specifying the mass density of the solid species and the mole fraction (stoichiometric coefficient) and name of each element in the solid species.

4.5.23 Fast Reactions Card

This card lists and describes all aqueous fast (equilibrium) reactions considered in the simulation. Fast reactions are defined by specifying the number of moles of each aqueous species participating in the reaction. The equilibrium coefficient must also be specified, either as a constant or as a polynomial function of temperature with a user-specified number of terms.

4.5.24 Kinetic Aqueous Reactions Card

This card lists and describes all kinetic reactions involving only aqueous species. An example of kinetic aqueous reactions is the biodegradation of dissolved organic carbon.

4.5.25 Slow Reactions Card

This card lists all kinetic reactions involving at least one solid species. Each reaction may involve one or two solid species. Examples of slow reactions include mineral dissolution or precipitation, metal corrosion, or biodegradation of soil organic matter. Slow reactions are defined by specifying the number of moles of each aqueous and solid species participating in the reaction. The equilibrium coefficient must also be specified, either as a constant or as a polynomial function of temperature, with a user-specified number of terms. Various parameters related to the rate law must also be defined. Three different types of rate law may be specified.

4.5.26 Fluid Decay Card

The option to simulate the radioactive decay of aqueous species is not currently implemented.

4.5.27 Solid Decay Card

The option to consider the radioactive decay of solid species is not currently implemented.

4.5.28 Lithology Card

This card specifies the relative fraction of each solid species found in each of the porous materials initially defined in the *Rock/Soil Zonation* Card Card, described in Section 4.5.4.

4.5.29 Reaction BC Card

This card lists the boundary conditions, specified at the upper boundary, for all aqueous and gas species. This simplification was originally introduced into the code because the upper boundary is usually the location of the inlet of a 1-dimensional column. In the future, the ability will be added to define specified concentration (Dirichlet) or specified flux (Neumann) conditions along any boundary segment of the domain.

4.5.30 Reaction IC Card

Currently, the only option is to set the initial aqueous species concentrations equal to those specified at the upper boundary.

5.0 Parameter Files

5.1 Introduction

This section describes the format and contents of the parameters file and the parameters.astct file, that are required to compile and execute the STORM simulator. Parameters are used by the FORTRAN programming 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 compilation time. No execution errors occur if the memory allocated is greater that required by the simulation, unless the memory requirements exceed the computer capabilities. Unless necessary, the user avoids executing simulations that 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 STORM simulator requires definitions of two types of parameters (declared and computed), 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 STORM simulator 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 following 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 one. Undersized parameters will generally yield execution errors that may or may not be detected by the system. Oversized parameters are permissible, but can result in excessive memory allocation.

5.2 Parameters File

5.2.1 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 seven lines of simulation notes and four execution periods would have a solution/simulation parameter set that appeared as

PARAMETER(LNOTES=7, LEPD=4)

5.2.2 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 r-, ` -, and z-directions, respectively, for cylindrical coordinate systems. These parameters 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 and 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 one node. The number of active dimensions is a number between one and three, inclusive. The parameter "LMNP" equals the minimum plane size and should equal the minimum of the following three products: "LFX*LFY," "LFY*LFZ," and "LFZ*LFX." A simulation with 2 active dimensions, having 22, 1, and 103 nodes in the x-, y-, and z- directions, respectively, and 51 inactive nodes has a computational domain parameter set that appeared as

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

5.2.3 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 one indicates the equation is solved and a value of zero indicates the equation is unsolved. The parameter "LFC" is a switch for freezing conditions. The parameter "LS" is a switch for coupled solute transport. A value of zero indicates nonfreezing conditions, whereas a value of one indicates freezing conditions. The switch combinations set with these parameters agrees with the operational mode. A simulation using the *Water-Air/Transport* operational mode has a operational mode parameter set that appeared as

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

5.2.4 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 one indicates the solver is active and value of zero 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, similar to a toggle, are the only means the user has for declaring a linear equation solution scheme. A simulation that used the banded matrix linear system solver has a linear system solver parameter set that appeared as

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

5.2.5 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 maximum 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 has a boundary parameter set that appeared as

PARAMETER(LBC=42, LBTM=26)

5.2.6 Source Parameters

The parameters "LSR" and "LSTM" equal the number of sources 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 one 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 3 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 appears as

PARAMETER(LSR=3, LSTM=26)

5.2.7 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 equals 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 has a unique name. The number of solutes should equal the number of solutes defined on the *Solute/Fluid Interactions* card. Each solute must have a unique name. A simulation with 14 rock/soil zonation domains, 3 rock/soil types, 7 solutes for the *Water-Air/Transport* operational mode has a rock types and solutes parameter set that appeared as

PARAMETER(LRC=3, LSOLU=7)

5.2.8 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 equals the *Number of Reference Nodes* input on the *Output Control* card. The number of *plot* file output times equals the *Number of Plot File Times* input on the *Output Control* card. The *plot* file that is generated, by default at the conclusion of an execution is not 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)

5.2.9 k-S-P Function Parameters

The parameters "LTBL" and "LPATH" equal the total number of table entries and the number of scanning paths for hysteretic saturation functions, respectively. The number of table entries equals the sum of *Number of Table Entries* on the *Saturation Function, Aqueous Relative Permeability Function*, and *Gas Relative Permeability Function* cards. The number of scanning paths equals the *Number of Scanning Paths* input for hysteretic saturation functions on the *Saturation Function* card. The recommended number of scanning paths for a hysteretic saturation function is seven. For non-hysteretic saturation functions, the number of scanning paths should equal one. 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 non-hysteretic 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 has a *k-S-P* function parameter set that appears as

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

5.2.10 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, LPH=LL+LG+LN)
PARAMETER(LHBW=LUK*LMNP+LUK-1)
PARAMETER(LCOL=LBD*LAN*LUK + LCG*(2*LAD+1))
PARAMETER(LROW=LBD*(3*LHBW+1) + LCG*LAN*LUK)
PARAMETER(LSCG=LBD + LCG*LAN*LUK)
PARAMETER(LSBD=LBD*LAN*LUK + LCG)
PARAMETER(LSOL=LAN*LUK)
PARAMETER(LSV=LUK+2, LSFV=2*LUK+1)
PARAMETER(LFD=LFX*LFY*LFZ)
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)
PARAMETER(LSXT=LSX**LT,LSXL=LSX**LL,LSXG=LSX**LG,LSXN=LSX**LN)
PARAMETER(LSXC=LSX**LC, LSXS=LSX**LS)
PARAMETER(LSYT=LSY**LT,LSYL=LSY**LL,LSYG=LSY**LG,LSYN=LSY**LN)
PARAMETER(LSYC=LSY**LC, LSYS=LSY**LS)
PARAMETER(LSZT=LSZ**LT,LSZL=LSZ**LL,LSZG=LSZ**LG,LSZN=LSZ**LN)
PARAMETER(LSZC=LSZ**LC, LSZS=LSZ**LS)
PARAMETER(LRCT=LRC**LT,LRCL=LRC**LL,LRCG=LRC**LG,LRCN=LRC**LN)
```

```
PARAMETER(LRCC=LRC**LC, LRCI=LRC**LFC, LRCS=LRC**LS)
PARAMETER(LBCT=LBC**LT,LBCL=LBC**LL,LBCG=LBC**LG,LBCN=LBC**LN)
PARAMETER(LBCC=LBC**LC, LBCI=LBC**LFC, LBCS=LBC**LS)
PARAMETER(LBCU=LUK+LPH, LBCV=LBCU+LSOLU, LOUPV=100+11*(LSOLU))
```

5.3 parameters.astct File

5.3.1 Specified Parameters

The parameter *maxxd* specifies the maximum number of points in the vertical (Z) direction, and is always set equal to the value of LFZ+1. The parameter *maxyd* specifies the maximum number of points in the horizontal (X) direction, and is always set equal to the value of LFX+1. The parameter *maxaqs* specifies the maximum number of aqueous species. The parameter *maxsld* specifies the maximum number of solid species. The parameter *maxfrxn* specifies the maximum number of fast reactions. The parameter *maxsrxn* specifies the maximum number of slow reactions. The parameter *maxfdec* specifies the maximum number of fluid decays. The parameter *maxsdec* specifies the maximum number of solid decays. The parameter *maxkrxn* specifies the maximum number of kinetic aqueous reactions. The parameter *maxlith* specifies the maximum number of lithologic units, and is always set equal to the value of LRC. The parameter *maxelem* specifies the maximum number of elements in one species. The parameter *maxgas* specifies the maximum number of gaseous species. An example of the specified parameters in a parameters.astct file is

```
parameter(maxxd=822,maxyd=26)
parameter(maxaqs=40,maxsld=20)
parameter(maxfrxn=20,maxsrxn=20)
parameter(maxfdec=1,maxsdec=1)
parameter(maxkrxn=1)
parameter(maxlith=6)
parameter(maxelem=14)
parameter(maxgas=3)
```

5.3.2 Computed Parameters

The parameter *maxrxn* specifies the maximum total number of reactions. The last line in the parameters.astct file is, therefore,

```
parameter(maxrxn=maxfrxn+maxkrxn+maxsrxn+maxfdec+maxsdec)
```

Table 5.1. Computed Parameter Definitions

Parameter	Definition
LUK	Number of solved coupled equations
LPH	Number of phases
LHBW	Block banded matrix half-band width
LCOL	Number of columns in the Jacobian matrix
LROW	Number of rows in the Jacobian matrix
LSCG	Number of unknowns for the conjugate gradient solver
LSBD	Number of unknowns for the banded matrix solver
LSOL	Number of unknowns
LSV	Number of field variable indexes
LSFV	Number of flux variable indexes
LFD	Number of nodes
LSX	Number of surfaces in the x- or r-directions
LSY	Number of surfaces in the y- or ÿ-directions
LSZ	Number of surfaces in the z-direction
LFDT	Number of energy field variables
LFDL	Number of aqueous field variables
LFDG	Number of gas field variables
LFDC	Number of solute transport equation field variables
LFDI	Number of solute transport equation field variables
LSXT	Number of energy flux variables in the x-direction
LSXL	Number of aqueous flux variables in the x-direction
LSXG	Number of gas flux variables in the x-direction
LSXC	Number of solute flux variables in the x-direction
LSYT	Number of energy flux variables in the y-direction
LSYL	Number of aqueous flux variables in the y-direction
LSYG	Number of gas flux variables in the y-direction
LSYC	Number of solute equation surfaces in the y-direction
LSZT	Number of energy flux variables in the z-direction
LSZL	Number of aqueous flux variables in the z-direction
LSZG	Number of gas flux variables in the z-direction
LSZC	Number of solute equation surfaces in the z-direction
LRCT	Number of energy variable rock/soil types
LRCL	Number of aqueous variable rock/soil types
LRCG	Number of gas variable rock/soil types
LRCC	Number of solute variable rock/soil types
LBCT	Number of energy boundary condition variables
LBCL	Number of aqueous boundary condition variables

Table 5.1. Computed Parameter Definitions (contd)

Parameter	**Definition**
LBCG	Number of gas boundary condition variables
LBCC	Number of solute boundary condition variables
LBCU	Number of coupled boundary condition variables
LBCV	Number of total boundary condition variables
LOUPV	Number of output variables

6.0 Compilation and Execution

6.1 Introduction

The STORM simulator 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, if the computer has sufficient memory. Although a general correspondence is present in computer design between memory size and execution speed, sufficient memory to compile and execute does not guarantee the machine has sufficient execution speeds to complete a simulation within a reasonable time period. The STORM simulator should be thought of as a collection of source code libraries. The libraries that become part of the executable code are 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 and parameters.astc files to define memory requirements, and linking the compiled object files to create an executable. An alternate option is to assemble or concatenate all of the source code libraries required for a particular operational mode into a single file and, subsequently, compile the assembled source code. Either procedure can be performed manually or automatically. The *make* utility (Talbott 1988) that 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 is designed to generate an executable or assembly source code for the STORM simulator.

6.2 Compilation

Executing the make command compiles and links all of the source and object coding required for an operational mode into a single executable file. Operational mode options are specified on the command line by specifying the name of the executable. As an example, the following UNIX shell command string on a computer that supports the *make* utility generates an optimized executable of the *Water-Air-Energy* operational mode

```
make storm3
```

where a file named "storm3" is generated. Table 6.1 shows the correspondence between the name of the executable and operational modes.

---

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

Table 6.1. Glossary of Operational Mode Indexes

Index	Operational Mode
storm1	*Water/Reactive Transport*
storm2	*Water-Air/Reactive Transport*
storm3	*Water-Air-Energy/Reactive Transport*

Assembly options are entered on a line within the file named *Makefile*, and assigned to a variable named FFLAGS. For example, the following line compiles the code on a SUN workstation with the options to generate a fast executable with the extended source code line length (132 characters) turned on.

```
FFLAGS = -fast -e
```

The assembly process has certain requirements with respect to source files, the commons file, the parameters file, and the *make* description file. Storing all of the source files (those files that have *.f* or *.c* extensions) and the commons file in a single directory is recommended. A path name to this directory can be defaulted into the make description file.

Executing make produces object files, with *.o* extensions for every required FORTRAN source file. These object files 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 automatically regenerates the object and executable files. Because of the updating features in make, any recompilation using a different assembly option requires the user to remove all of the object files from the current directory.

6.3 Execution

Executing the STORM simulator is straightforward and only requires that an input file named *input* resides 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 path to and the name of the executable file. Execution is indicated by printing a STORM title banner and program disclaimer to the standard input/output device (for example, a monitor).

6.3.1 Error Messages

Two types of error messages can be generated during a STORM execution. The first type is a system-generated message that typically indicates a memory error, FORTRAN error, or other system error identified by the system. The second type of error messages is that generated by the STORM code that typically refer to input, parameter, or convergence failure type messages. STORM-generated messages are divided into three categories, according to severity. The most severe are the ERROR messages that abort the program execution. Undersized parameters are typical of errors that yield

ERROR messages because execution of the simulator with undersized parameters may yield gross errors, or even worse, subtle errors that can pass undetected in the results. Next on the severity level are the WARNING messages that generally indicates to the user that a problem with the input file probably exists. The least severe are NOTE messages that are used to record events such as the absence of an optional input card.

When an execution is terminated with a STORM-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. To a limited degree, the STORM simulator also traps input formatting errors. Unfortunately, the resulting input error message that results may not always agree with the actual location of the identified input error. This situation arises because the original error may have skipped detection. For example, a missing input item on an input line is not identified until STORM identifies an incorrect data type or the end of the line was reached. The user will 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 that 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 appears as

/STORM1/RDINPT1/RDRPL/WRMSG

where the called routine names are delineated with a forward slash. For this example, the main routine STORM1 called the subroutine RDINPT1 to read the input file for the *Water* operational mode. It in turn, called RDRPL, the global routine, to read the *Aqueous Relative Permeability* card that identified an error and called WRMSG to write an error message to the standard input/output device.

7.0 Output Files

7.1 Introduction

The STORM 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. If the simulation concludes without a fatal error, a *restart* and *plot* file are also generated. *Restart* and *plot* files names include an extension that corresponds to the time step at which the file was written. For example, a restart file named *restart.39* was written at the conclusion of the 39th time step. Simulations involving hysteretic saturation functions also produce a *hyster* file that contains saturation history records. *Hyster* file names also include an extension that corresponds to the time step at which the file was written. A *surface* file is only generated when a *Surface Flux* card is included in an input file. All output files generated by the STORM simulator are ASCII text files.

7.2 Output File

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

```
Welcome to ...

                    STORM
   Subsurface Transport Over Reactive Multiphases

This file was produced by STORM, a numerical simulator
developed by the Pacific Northwest National Laboratory
with support from the U.S. Department of Energy.

Results from this version of STORM should not be used
for license related applications. Distribution of the
STORM source code is strictly prohibited.

For inquiries or assistance:  Call (509) 372-6132

                  --- OUTPUT ---
```

Following this banner is an input record that documents the simulation input. An input record is included on the *output* file to document the simulation in the event 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 that are not included in the *input* file are noted in the input record. If an input error is identified by the STORM simulator, an error message appears in the input record at the point the error was noted. Input cards are read by the STORM simulator in a specific order and appear in that order in the input record. 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.

7.3 Refnod File

The *refnod* (reference node) output file is a table of simulation data and selected reference node variables that are printed according the frequency requested on the *Output Control* card with the input item *Reference Node Output File Frequency*. Every 10 print records a table column header is printed to delineate the columns. Each print record shows 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 indexes 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 indexes of 2, 3, and 6, respectively, has 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 a reference node output record for a horizontal heat pipe problem, involving 50 nodes for time steps 70 through 73, appears 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
```

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

7.4 Plot File

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

```
Welcome to ...


                    STORM
     Subsurface Transport Over Reactive Multiphases


This file was produced by STORM, a numerical simulator
developed by the Pacific Northwest National Laboratory
with support from the U.S. Department of Energy.

Results from this version of STORM should not be used
for license related applications. Distribution of the
STORM source code is strictly prohibited.

For inquiries or assistance: Call (509) 372-6132

                  --- 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* 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 that 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 file and contain the x- or r-direction node positions, y- or θ-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 or fluxes are written twice to the plot file. 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.

Two post-processing programs are available, d_plot1d and d_plot2d. The program d_plot1d parses output data from a 1-dimensional simulation into columnar format. The program d_plot2d parses data from a 2-dimensional simulation into Spyglass Transform format.

7.5 Surface File

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

Welcome to ...

STORM
Subsurface Transport Over Reactive Multiphases

This file was produced by STORM, a numerical simulator
developed by the Pacific Northwest National Laboratory
with support from the U.S. Department of Energy.

Results from this version of STORM should not be used
for license related applications. Distribution of the
STORM source code is strictly prohibited.

For inquiries or assistance: Call (509) 372-6132

--- SURFACE ---

Following the header, are columns of surface flux rate and integral data. Surface flux information is written to the *surface* file at the conclusion of every time step. The first column of data in the *surface* file is the time step in units that 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 is the surface flux rate at the current time step and the column is 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 under the column title. Definitions of the header abbreviations for the *surface* file are given in Table 7.1. 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 (that is, the surface flux definition order).

7.6 Restart File

A *restart* file is generated by default at the conclusion of every execution and also when requested on the *Output Control* card with the *Plot File Output Times* input items. If a file named *restart.xx* (where *xx* refers to the time step extension) already exists in the current directory, the file is erased and overwritten.

Table 7.1. 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)
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)
UQV	Energy flux rate (x-direction)
UQVS	Energy 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)
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)
VQV	Energy flux rate (y-direction)
VQVS	Energy 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)
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)
WQV	Energy flux rate (z-direction)
WQVS	Energy flux integral (z-direction)

The user is responsible for renaming previous restart.xx files prior to executing the STORM simulator. A *restart* file begins with a welcome statement, a disclaimer, and a banner that appear similar to

```
Welcome to ...


                    STORM
   Subsurface Transport Over Reactive Multiphases


This file was produced by STORM, a numerical simulator
developed by the Pacific Northwest National Laboratory
with support from the U.S. Department of Energy.

Results from this version of STORM should not be used
for license related applications. Distribution of the
STORM source code is strictly prohibited.

For inquiries or assistance:  Call (509) 372-6132

                    --- 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 that contain the following field variable data for the current time step for every node (one line per node): temperature, aqueous pressure, gas pressure, water vapor pressure, aqueous saturation, and gas saturation. If the simulation includes solute transport, the node data lines also contain solute volumetric concentrations for each solute. Moreover, if the simulation includes freezing water conditions, the node data lines 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 Manual (White and Oostrom 1996) for a discussion of phase conditions.

8.0 Model Verification

8.1 Aqueous Phase Advection and Diffusion

For the case of 1-dimensional, steady state diffusion and advection of a single aqueous species, with constant production from a single reaction with depth, Equations (2.17) through (2.27)simplify to

$$W = -\left(\theta_a \mathbf{D}^a\right)\frac{\partial^2 C^a}{\partial z^2} + \left(\theta_a \mathbf{V}_a\right)\frac{\partial C^a}{\partial z} \tag{8.1}$$

Assuming production occurs over a finite depth 0<*z*<*L*, with a specified concentration (C_0^a) at the top boundary (*z* = 0) and no diffusive flux at the lower boundary (z = L), the aqueous phase concentration of that gas species is given by

$$C_z^a = \frac{\mathbf{D}\left(e^{\frac{L\mathrm{v}}{\mathbf{D}}} - e^{\frac{(L-z)\mathrm{v}}{\mathbf{D}}}\right)W + \mathrm{v}\left(C_0^a \mathrm{v} - Wz\right)}{\mathrm{v}^2} \tag{8.2}$$

where $\mathbf{V} = \theta_a \mathbf{V}_a$ and $\mathbf{D} = \theta_a \mathbf{D}^a$. A STORM simulation of Halite dissolution was run that corresponds to this analytical solution. Both the *Fully Coupled* and the *Operator Split* solution modes were used. Output from STORM was compared to the analytical solution. The model parameters are listed in Table 8.1; the results are shown in Figure 8.1. Results for STORM run in the *Fully Coupled* solution mode and the analytical solution agree well. Results for STORM run in the *Operator Split* mode agree; if the time step is constrained to a maximum value of 0.5 hour.

Table 8.1. Simulation Parameters for Gas Phase Diffusion

Parameter	**Value**
Temperature, C	25
C_0^a, mol kg^{-1}	1.0x10^{-7}
D^a, m^2 s^{-1}	5.0x10^{-4}
V_a, m s^{-1}	1.6x10^{-8}
θ_a	5.2x10^{-2}
W, mol m^{-3} s^{-1}	1.2x10^{-9}

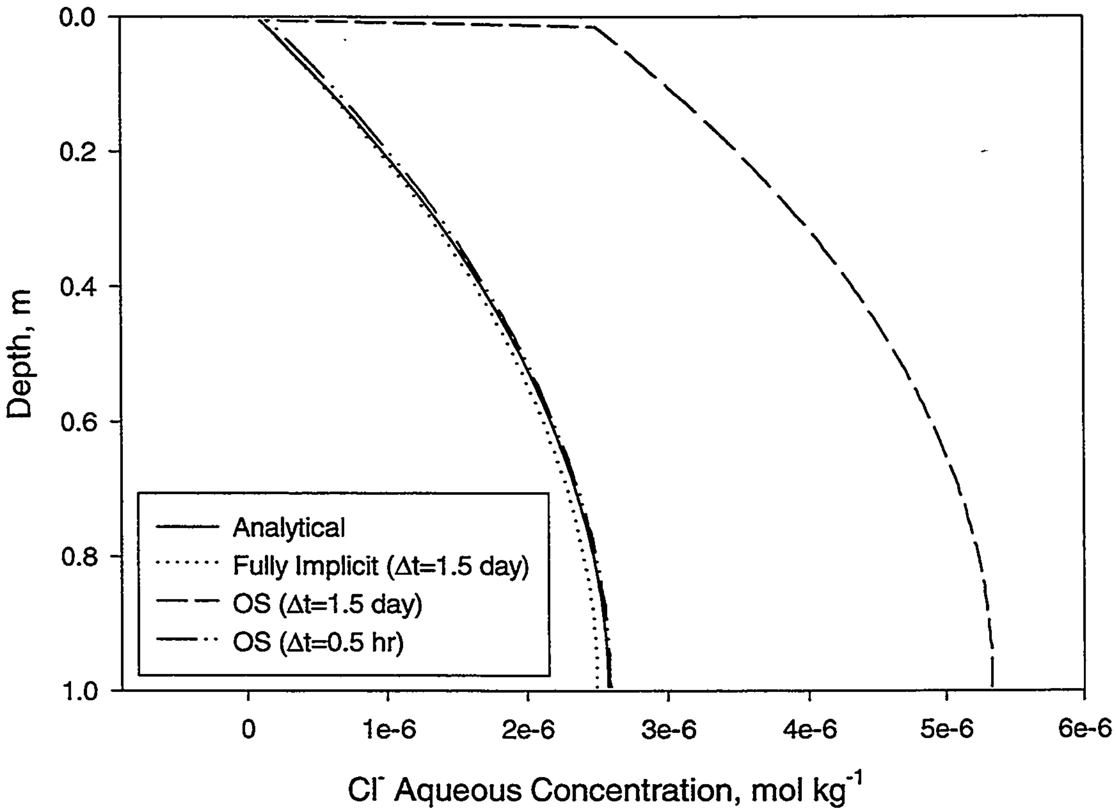

Figure 8.1. Comparison of STORM Results Using Either *Fully Coupled* or *Operator Split* Solution Modes to Analytical Solution for Constant Source, Advection, and Diffusion

Data File 8.1. Input File for Aqueous Phase Advection and Diffusion Test Case

```
~Simulation Title Card
1,
Organic matter decomposition,
DH Bacon,
Pacific Northwest National Laboratory,
August 1998,
00:00:00 AM PDT,
0,

~Solution Control Card
Restart,
Water w/ Transport,
1,
0.0,hr,1000.,yr,1.,s,5.,yr,1.005,8,1.0e-6,
1.e+30,yr,1.e+30,yr,100000,
0,
```

```
~Grid Card
Uniform Cartesian,
1,1,100,
1.0,cm,
1.0,cm,
1.0,cm,

~Rock/Soil Zonation Card
2,
TopSoil,1,1,1,1,1,50,
BottomSoil,1,1,1,1,51,100,

~Mechanical Properties Card
TopSoil,2.72,g/cm^3,0.371,0.371,1.e-5,1/m,Millington and Quirk,
BottomSoil,2.72,g/cm^3,0.371,0.371,1.e-5,1/m,Millington and Quirk,

~Hydraulic Properties Card
TopSoil,9467.28,hc m/yr,,,9467.28,hc m/yr,Constant,
BottomSoil,9467.28,hc m/yr,,,9467.28,hc m/yr,Constant,

~Saturation Function Card
TopSoil,Nonhysteretic van Genuchten,6.83,1/m,2.08,0.1213,,
BottomSoil,Nonhysteretic van Genuchten,6.83,1/m,2.08,0.1213,,

~Aqueous Relative Permeability Card
TopSoil,Mualem,,
BottomSoil,Mualem,,

~Gas Relative Permeability Card
TopSoil,Mualem,,
BottomSoil,Mualem,,

~Boundary Conditions Card
2,
Top,Neumann Aqueous,Initial Cond,Initial Cond,Initial Cond,Initial Cond,Initial
Cond,Initial Cond,
1,1,1,1,100,100,1,
0.0,hr,-0.05,cm/yr,,,,,,,,,,,,,
Bottom,Free Gradient,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,
1,1,1,1,1,1,1,
0.0,hr,,,,,,,,,,,,,,,

~Source Card
1,
Aqueous Mass,1,1,1,1,1,100,1,
0.,s,0.,kg/s,,

~Output Control Card
1,
1,1,100,
1,1,hr,cm,3,5,5,
```

```
0,
14,
0.0,yr,
1.0,min,
1.0,day,
1.0,wk,
4.3,wk,
1.0,yr,
2.0,yr,
5.0,yr,
10.0,yr,
50.0,yr,
100.0,yr,
500.0,yr,
1000.0,yr,
5000.0,yr,
14,
Gas Pressure,Pa,
Aqueous Pressure,Pa,
Gas Saturation,,
Aqueous Saturation,,
Aqueous Density,kg/m^3,
Rock/Soil Type,,
Diffusive Porosity,,
Aqueous Relative Permeability,,
z Aqueous Volumetric Flux,m/s,
Solute Aqueous Conc,H+,1/m^3,
Solute Aqueous Conc,H2O,1/m^3,
Solute Aqueous Conc,OH-,1/m^3,
Solute Aqueous Conc,Na+,1/m^3,
Solute Aqueous Conc,Cl-,1/m^3,

~Surface Flux Card
1,
Aqueous Volumetric Flux,m^3/s,m^3,Top,1,1,1,1,100,100,

~Reactive Transport Control card
Normal,
Fully Coupled,
0,1,0,1,1,0,0,0,0,
 2.00000E-06, 1.00000E+00, 1.00000E-06, 1.00000E-07, 1.00000E-06, 5.00000E-03,
1.00000E-06,

~Solute/Fluid Interaction Card
5,
H+,Constant,5.0e-4,m^2/s,Continuous,1.e20,yr,
H2O,Constant,5.0e-4,m^2/s,Continuous,1.e20,yr,
OH-,Constant,5.0e-4,m^2/s,Continuous,1.e20,yr,
Na+,Constant,5.0e-4,m^2/s,Continuous,1.e20,yr,
Cl-,Constant,5.0e-4,m^2/s,Continuous,1.e20,yr,
0,
```

```
~Solute/Porous Media Interaction Card
TopSoil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
Na+,0.0,m^3/kg,
Cl-,0.0,m^3/kg,
BottomSoil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
Na+,0.0,m^3/kg,
Cl-,0.0,m^3/kg,

~Aqueous Species card
 5,
'H+          ', 1.0000, 9.0000, 1,
 1.0000,' H',
'H2O         ', 0.0000,-4.0000, 2,
 2.0000,' H',
 1.0000,' O',
'OH-         ',-1.0000, 3.0000, 2,
 1.0000,' H',
 1.0000,' O',
'Na+         ', 1.0000, 4.0000, 1,
 1.0000,'Na',
'Cl-         ',-1.0000, 3.0000, 1,
 1.0000,'Cl',

~Gas Species card
   0,

~Components card
   0,

~Solid Species card
  1,
'Halite', 2.1633,  2,
 1.0000,'Na',
 1.0000,'Cl',

~Fast Reactions card
   1,
   3,
-1.0000,'H2O         ',
 1.0000,'H+          ',
 1.0000,'OH-         ',
   5,    1.00000,
-1.49400E+01, 4.17100E-02,-1.91700E-04, 4.95000E-07,-6.15900E-10,
```

```
~Kinetic Aqueous Reactions card
   0,

~Slow Reactions card
 1,
 0, 1,
-1.0000,'Halite',
   2,
 1.00000E+00,'Na+         ', 1.0000, 1.0000,
 1.00000E+00,'Cl-         ', 1.0000, 1.0000,
    -22.2669,  0.,    1.00000,  0.0, 1,
1.5855,

~Fluid Decay card
   0,

~Solid Decay card
   0,

~Lithology card
'top soil    ', 1,
'Halite', 1.00000E+00, 0.50000E-02,
'bottom soil ', 1,
'Halite', 1.00000E+00, 0.50000E-02,

~Reaction BC card
   1,
1,1,
   0,
   0,
   0,
   4,
  0.10000E+01,  1,
 1.0000,'H2O          ',
  0.10000E-06,  1,
 1.0000,'H+           ',
  0.10000E-06,  1,
 1.0000,'Na+          ',
  0.10000E-06,  1,
 1.0000,'Cl-          '

~Reaction IC card
   1,
```

8.2 Gas Phase Advection and Diffusion

For the case of 1-dimensional, steady state diffusion of a single gas species, with constant production from a single reaction with depth, Equation (2.17) simplifies to

$$W = -\left(\theta_g \mathbf{D}^g K_H'^{-1} + \theta_a \mathbf{D}^a\right)\frac{\partial^2 C^a}{\partial z^2} \quad . \qquad (8.3)$$

Assuming production occurs over a finite depth $0<z<L$, with a specified concentration (C_0^a) at the top boundary ($z = 0$) and no diffusive flux at the lower boundary ($z = L$), the aqueous phase concentration of that gas species is given by

$$C_z^a = C_0^a + \frac{W}{\left(\theta_g \mathbf{D}^g K_H'^{-1} + \theta_a \mathbf{D}^a\right)}\left(zL - \frac{z^2}{2}\right) \quad . \qquad (8.4)$$

and the gas phase concentration may be determined using Equation (2.16). A STORM simulation was run that corresponds to this analytical solution. The model parameters are listed in Table 8.2.

Results for simulations with two different $CO_2(g)$ production rates are shown in Figure 8.2 (open and closed diamond symbols). Results for STORM and the analytical solution agree quite well. Concentrations of $CO_2(g)$ shown are reasonable values found in vadose zones where organic matter is being decomposed by aerobic bacteria.

Table 8.2. Simulation Parameters for Gas Phase Diffusion

Parameter	**Value**
Temperature, C	15
C_0^a, atm	$3\text{x}10^{-4}$
K_H'	1.088
D^g, $m^2\ s^{-1}$	$1.6\text{x}10^{-6}$
D^a, $m^2\ s^{-1}$	$5\text{x}10^{-9}$
θ_g	0.24
θ_a	0.0266
W, mol $m^{-3}\ s^{-1}$	$4.65\text{x}10^{-9}$ or $9.60\text{x}10^{-9}$

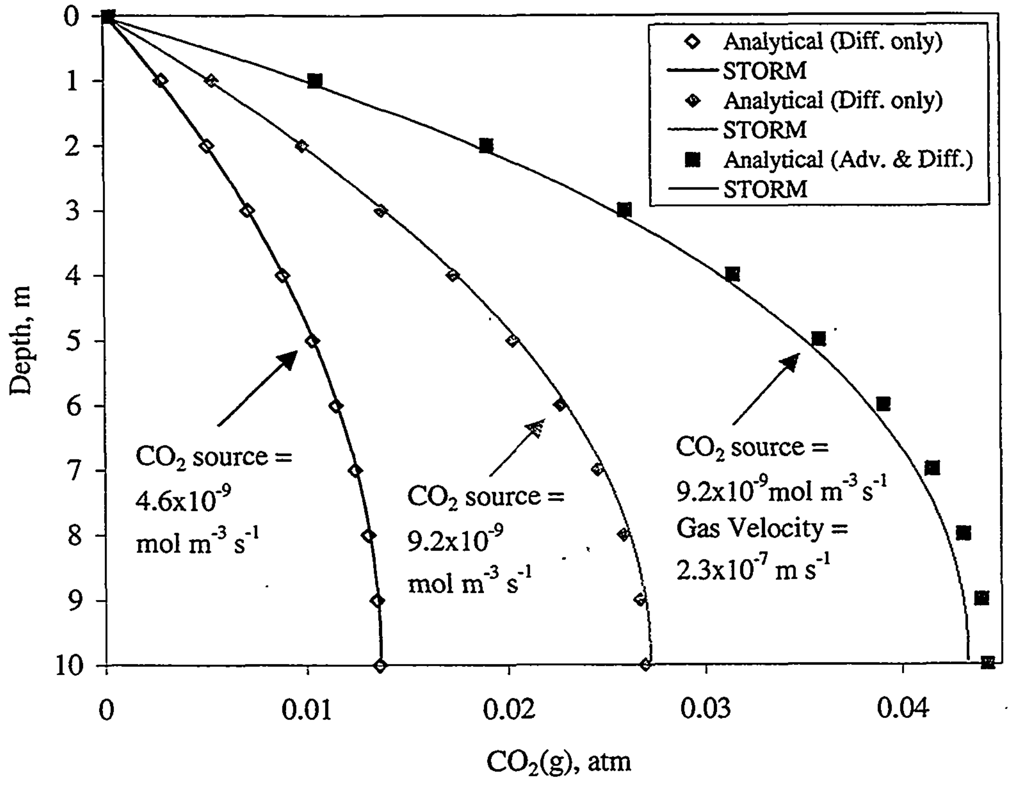

Figure 8.2. Results of Gas Phase Diffusion and Advection Test Cases, Comparing STORM with Analytical Solutions for Constant Gas-Phase Production and Diffusion, with and without Advection

Data File 8.2. STORM Input Data File for Gas Phase Advection and Diffusion Test Case

```
~Simulation Title Card
1,
Organic matter decomposition,
DH Bacon,
Pacific Northwest National Laboratory,
August 1998,
00:00:00 AM PDT,
1,
With O2(g) and CO2(g) diffusion

~Solution Control Card
Restart w/ no flow,
Water w/ air w/ transport,
1,
0.0,hr,1000.,yr,1.,min,5.,yr,1.005,8,1.0e-6,
1.e+30,yr,1.e+30,yr,100000,
Variable Aqueous Diffusion,
```

```
Variable Gas Diffusion,
1,
Aqueous Relative Permeability,Geometric,

~Grid Card
Uniform Cartesian,
1,1,100,
1.0,cm,
0.1,cm,
10.0,cm,

~Rock/Soil Zonation Card
2,
TopSoil,1,1,1,1,1,50,
BottomSoil,1,1,1,1,51,100,

~Mechanical Properties Card
TopSoil,2650.0,kg/m^3,0.2667,0.2667,,,Millington and Quirk,
BottomSoil,2650.0,kg/m^3,0.2667,0.2667,,,Millington and Quirk,

~Hydraulic Properties Card
TopSoil,,,,,3.156e-5,hc cm/s,Constant,
BottomSoil,,,,,3.156e-5,hc cm/s,Constant,

~Saturation Function Card
TopSoil,Nonhysteretic van Genuchten,0.044,1/cm,2.2,0.08494,,
BottomSoil,Nonhysteretic van Genuchten,0.044,1/cm,2.2,0.08494,,

~Aqueous Relative Permeability Card
TopSoil,Tabular,33,
3.5026E-09,3.1593E-07,
0.01225919,6.7082E-07,
0.02977233,1.7099E-06,
0.04728546,3.8458E-06,
0.0647986,7.8611E-06,
0.08231173,1.4901E-05,
0.117338,4.5067E-05,
0.15236427,0.00011488,
0.18739054,0.00025837,
0.22241681,0.00052812,
0.25744308,0.00100111,
0.29246935,0.0017854,
0.32749562,0.00302769,
0.36252189,0.00492173,
0.39754816,0.00771754,
0.43257443,0.01173155,
0.4676007,0.01735761,
0.50262697,0.02507879,
0.53765324,0.03548017,
0.57267951,0.04926245,
0.60770578,0.06725641,
```

```
0.64273205,0.09043837,
0.67775832,0.11994642,
0.71278459,0.15709761,
0.74781086,0.20340604,
0.78283713,0.26060184,
0.8178634,0.330651,
0.85288967,0.41577624,
0.88791594,0.51847864,
0.92294221,0.64156028,
0.95796848,0.78814774,
0.99299475,0.96171657,
1,1,
BottomSoil,Tabular,33,
3.5026E-09,3.1593E-07,
0.01225919,6.7082E-07,
0.02977233,1.7099E-06,
0.04728546,3.8458E-06,
0.0647986,7.8611E-06,
0.08231173,1.4901E-05,
0.117338,4.5067E-05,
0.15236427,0.00011488,
0.18739054,0.00025837,
0.22241681,0.00052812,
0.25744308,0.00100111,
0.29246935,0.0017854,
0.32749562,0.00302769,
0.36252189,0.00492173,
0.39754816,0.00771754,
0.43257443,0.01173155,
0.4676007,0.01735761,
0.50262697,0.02507879,
0.53765324,0.03548017,
0.57267951,0.04926245,
0.60770578,0.06725641,
0.64273205,0.09043837,
0.67775832,0.11994642,
0.71278459,0.15709761,
0.74781086,0.20340604,
0.78283713,0.26060184,
0.8178634,0.330651,
0.85288967,0.41577624,
0.88791594,0.51847864,
0.92294221,0.64156028,
0.95796848,0.78814774,
0.99299475,0.96171657,
1,1,

~Gas Relative Permeability Card
TopSoil,Tabular,33,
1,1,
0.98774081,0.99999139,
```

```
0.97022767,0.99994918,
0.95271454,0.99987151,
0.9352014,0.99975797,
0.91768827,0.99960795,
0.882662,0.99919528,
0.84763573,0.99862545,
0.81260946,0.99788751,
0.77758319,0.99696733,
0.74255692,0.99584706,
0.70753065,0.99450437,
0.67250438,0.99291148,
0.63747811,0.99103385,
0.60245184,0.98882835,
0.56742557,0.98624081,
0.5323993,0.98320258,
0.49737303,0.97962579,
0.46234676,0.9753964,
0.42732049,0.97036419,
0.39229422,0.96432763,
0.35726795,0.95701052,
0.32224168,0.94802483,
0.28721541,0.93680881,
0.25218914,0.92251973,
0.21716287,0.90383668,
0.1821366,0.87857318,
0.14711033,0.84284783,
0.11208406,0.78910152,
0.07705779,0.70059708,
0.04203152,0.53266493,
0.00700525,0.13244732,
0,0,
BottomSoil,Tabular,33,
1,1,
0.98774081,0.99999139,
0.97022767,0.99994918,
0.95271454,0.99987151,
0.9352014,0.99975797,
0.91768827,0.99960795,
0.882662,0.99919528,
0.84763573,0.99862545,
0.81260946,0.99788751,
0.77758319,0.99696733,
0.74255692,0.99584706,
0.70753065,0.99450437,
0.67250438,0.99291148,
0.63747811,0.99103385,
0.60245184,0.98882835,
0.56742557,0.98624081,
0.5323993,0.98320258,
0.49737303,0.97962579,
0.46234676,0.9753964,
```

```
0.42732049,0.97036419,
0.39229422,0.96432763,
0.35726795,0.95701052,
0.32224168,0.94802483,
0.28721541,0.93680881,
0.25218914,0.92251973,
0.21716287,0.90383668,
0.1821366,0.87857318,
0.14711033,0.84284783,
0.11208406,0.78910152,
0.07705779,0.70059708,
0.04203152,0.53266493,
0.00700525,0.13244732,
0,0,

~Boundary Conditions Card
2,
Top,Neumann Aqueous,Neumann Gas,Initial Cond,Initial Cond,Initial Cond,Initial
Cond,Initial Cond,Initial Cond,
1,1,1,1,100,100,1,
0.0,hr,-8.4333E-12,m/s,,5.5e-08,m/s,1.,,,,,,,,,,,,,
Bottom,Dirichlet Aqueous,Dirichlet
Gas,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,
1,1,1,1,1,1,1,
0.0,hr,3.3100E+04,Pa,,101325.0,Pa,1.,,,,,,,,,,,,,

~Source Card
1,
Aqueous Mass,1,1,1,1,1,100,1,
0.,s,60000.,Pa,0.,kg/s,,
Gas Mass,1,1,1,1,1,100,1,
0.,s,60000.,Pa,0.,kg/s,,

~Output Control Card
1,
1,1,100,
1,1,hr,cm,3,5,5,
0,
14,
0.0,yr,
1.0,min,
1.0,day,
1.0,wk,
4.3,wk,
1.0,yr,
2.0,yr,
5.0,yr,
10.0,yr,
50.0,yr,
100.0,yr,
500.0,yr,
```

```
1000.0,yr,
5000.0,yr,
11,
Gas Pressure,Pa,
Aqueous Pressure,Pa,
Gas Saturation,,
Aqueous Saturation,,
Aqueous Density,kg/m^3,
Gas Density,kg/m^3,
Rock/Soil Type,,
Diffusive Porosity,,
Aqueous Relative Permeability,,
z Aqueous Volumetric Flux,m/s,
z Gas Volumetric Flux,m/s,

~Surface Flux Card
1,
Aqueous Volumetric Flux,m^3/s,m^3,Top,1,1,1,1,100,100,

~Reactive Transport Control card
Normal,
Fully Coupled,
0,0,0,0,1,0,0,0,0,
 2.00000E-06, 1.00000E+00, 1.00000E-06, 1.00000E-07, 1.00000E-06, 5.00000E-03,
1.00000E-06,

~Aqueous Species card
 6,
'H+          ', 1.0000, 9.0000, 1,
 1.0000,' H',
'H2O         ', 0.0000,-4.0000, 2,
 2.0000,' H',
 1.0000,' O',
'OH-         ',-1.0000, 3.0000, 2,
 1.0000,' H',
 1.0000,' O',
'CO2(aq)     ', 0.0000, 0.0000, 2,
 1.0000,' C',
 2.0000,' O',
'O2(aq)      ', 0.0000, 3.0000, 1,
 2.0000,' O',
'Cl-         ',-1.0000, 3.0000, 1,
 1.0000,'Cl',

~Solute/Fluid Interaction Card
6,
H+,5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
H2O,5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
OH-,5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
CO2(aq),5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
O2(aq),5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
```

```
Cl-,5.0e-9,m^2/s,0.,m^2/s,Constant,0.,m^3/m^3,Continuous,1.e20,yr,
0,

~Solute/Porous Media Interaction Card
TopSoil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
CO2(aq),0.0,m^3/kg,
O2(aq),0.0,m^3/kg,
Cl-,0.0,m^3/kg,
BottomSoil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
CO2(aq),0.0,m^3/kg,
O2(aq),0.0,m^3/kg,
Cl-,0.0,m^3/kg,

~Gas Species card
   2,
'CO2(g)      ', 'CO2(aq)     ', 'diffusion', 1.69443E-06, 2,
 1.0000,' C',
 2.0000,' O',
 4,
-1.096100E+00,-1.796793E-02, 1.321931E-04,-3.982381E-07,
'O2(g)       ', 'O2(aq)      ', 'diffusion', 2.40000E-06, 1,
 2.0000,' O',
 4,
-2.656700E+00,-1.224250E-02, 1.117390E-04,-3.440481E-07,

~Components card
   0,

~Solid Species card
  1,
'Organic Matter', 2.5200,  3,
 1.0000,' C',
 2.0000,' H',
 1.0000,' O',

~Fast Reactions card
   1,
   3,
-1.0000,'H2O        ',
 1.0000,'H+         ',
 1.0000,'OH-        ',
   5,    1.00000,
-1.49400E+01, 4.17100E-02,-1.91700E-04, 4.95000E-07,-6.15900E-10,

~Kinetic Aqueous Reactions card
```

```
   0,

~Slow Reactions card
 1,
 0, 1,
-1.0000,'Organic Matter',
   3,
-3.60900E-02,'O2(aq)      ', 0.0000, 0.0000,
 1.08800E+00,'CO2(aq)     ', 0.0000, 0.0000,
 0.00000E+00,'Cl-         ', 1.0000, 1.0000,
    -22.2669,  0.,    1.00000,  0.0, 1,
-1.00000E+00,

~Fluid Decay card
   0,

~Solid Decay card
   0,

~Lithology card
'top soil    ', 1,
'Organic Matter', 1.00000E+00, 0.50000E-02,
'bottom soil ', 1,
'Organic Matter', 1.00000E+00, 0.50000E-02,

~Reaction BC card
   1,
1,1,
   0,
   0,
   2,
'CO2(g)       ', 3.00000E-04,
'O2(g)        ', 2.10000E-01,
   3,
  0.10000E+01,  1,
 1.0000,'H2O         ',
  0.10000E-06,  1,
 1.0000,'H+          ',
  2.15331E-03,  1,
 1.0000,'Cl-              '

~Reaction IC card
   1,
```

For the case of 1-dimensional, steady state diffusion and advection of a single gas species, with constant production from a single reaction with depth, Equation (2.17) simplifies to

$$W = -\left(\theta_g \mathbf{D}^g K_H^{'-1} + \theta_a \mathbf{D}^a\right)\frac{\partial^2 C^a}{\partial z^2} + \left(\theta_g \mathbf{V}_g K_H^{'-1} + \theta_a \mathbf{V}_a\right)\frac{\partial C^a}{\partial z} \tag{8.5}$$

Assuming production occurs over a finite depth $0<z<L$, with a specified concentration (C_0^a) at the top boundary ($z = 0$) and no diffusive flux at the lower boundary ($z = L$), the aqueous phase concentration of that gas species is given by

$$C_z^a = \frac{\mathbf{D}\left(e^{\frac{L\mathbf{v}}{\mathbf{D}}} - e^{\frac{(L-z)\mathbf{v}}{\mathbf{D}}}\right)W + \mathbf{v}\left(C_0^a\mathbf{v} - Wz\right)}{\mathbf{v}^2} \tag{8.6}$$

where $\mathbf{V} = \theta_g \mathbf{V}_g K_H^{'-1} + \theta_a \mathbf{V}_a$ and $\mathbf{D} = \theta_g \mathbf{D}^g K_H^{'-1} + \theta_a \mathbf{D}^a$. The gas phase concentration may be determined using Equation (2.16). A STORM simulation was run that corresponds to this analytical solution. The model parameters are listed in Table 8.2. The results are shown in green in Figure 8.2. Results for STORM and the analytical solution again agree quite well. The upward gas velocity of 2.3×10^{-7} m s^{-1} increases the CO_2(g) concentrations found at any given depth, as compared to a similar test case without gas advection (Figure 8.2, square symbols).

8.3 Gas Phase Mass Balance

A simple simulation runs where oxygen is consumed in one kinetic reaction. Hand calculations are used to verify that oxygen consumption rate is passed correctly to the flow routines and that overall gas mass balance is preserved. A 10-year simulation with a constant sink term for O_2(g) is used to test whether gas consumption rates are passed correctly from the transport subroutines to the flow subroutines, and whether the total air mass is conserved.

The O_2(g) consumption rate was specified as a constant -4.04×10^{-9} mol m^{-3} s^{-1}. In the STORM subroutine D_WATERSINK that calculates the water and air consumption rates passed to the flow routines, the total mass consumption rate of air in units of kg s^{-1} is calculated using

$$R_{\mathrm{Air}} = R_{\mathrm{O}_2} M_{\mathrm{O}_2} V_{\mathrm{node}} \tag{8.7}$$

where R_{O_2} is the O_2(g) consumption rate in mol m^{-3} s^{-1}, M_{O_2} is the molecular weight of oxygen in kg mol^{-1}, and V_{node} is the node volume in m^3. Using Equation (8.7), and assuming a molecular weight of 3.2×10^{-2} kg mol^{-1} for O_2(g) and a node volume of 1×10^{-6} m^3, the calculated R_{Air} should be

$$R_{\mathrm{Air}} = \left(-4.04\times10^{-9}\mathrm{mol\ m^{-3}s^{-1}}\right)\left(3.2\times10^{-2}\mathrm{kg\ s^{-1}}\right)\left(1\times10^{-6}\mathrm{m^3}\right) = 1.29\times10^{-16}\mathrm{kg\ s^{-1}}. \tag{8.8}$$

As shown in Table 8.3, this value agrees with that calculated by the transport subroutine D_WATERSINK.

Table 8.3. STORM Input/Output Variables for Test Case 3

Variable	Where Calculated	Initial	10 Years
Total Air Mass, kg	Flow subroutine	2.9366E-07	2.5289E-07
Air Mass Source Integral, kg	Flow subroutine	0.0000E+00	-4.0761E-08
Air Rate, kg s^{-1}	Transport subroutine	-1.2917E-16	-1.2915E-16
$O_2(g)$ Rate, mol m^{-3} s^{-1}	Transport subroutine	-4.0368E-09	-4.0362E-09

Over a period of 10 years, or $3.156\text{x}10^8$ s, the total mass of air consumed is equal to $(-1.2916\text{x}10^{-16}$ kg $s^{-1})(3.156\text{x}10^8$ s) = $-4.076\text{x}10^{-7}$ kg. This figure is equal to the Air Mass Source Integral calculated by the flow subroutines, indicating the Air Mass Source Rate is being passed correctly from the transport subroutines to the flow subroutines. The difference of the Total Air Mass between the beginning and the end of the simulation is equal to $2.5289\text{x}10^{-7}$ kg – $2.9366\text{x}10^{-7}$ kg = $-4.077\text{x}10^{-7}$ kg, indicating that total air mass is being conserved. The small error in these calculations is due to the small decrease in R_{O_2} over the course of the simulation. As the solid (organic matter) decay reaction responsible for the consumption of $O_2(g)$ proceeds, the surface area of organic matter decreases slightly, thus lowering R_{O_2}.

8.4 Quartz Dissolution

The chemical reaction solver of STORM is verified by comparing STORM results to EQ3/6. In STORM, advective and diffusive transport were made negligible by assuming a very low hydraulic conductivity for the porous media and very small aqueous diffusion coefficients for aqueous species. As an example of a simple kinetic reaction path, quartz sand reacts at 100°C with deionized water according to the reaction

$$\text{Quartz} \rightleftharpoons \text{SiO}_2(aq) \tag{8.9}$$

Quartz dissolves according to the pH-independent rate law (Rimstidt and Barnes 1980)

$$r_{qtz} = A_s k\left(1-\frac{Q}{K}\right) \tag{8.10}$$

where r_{qtz} is the dissolution rate (mol s^{-1}), A_s is the surface area of the mineral (500 m^2), k is the reaction rate constant ($2\text{x}10^{-11}$ mol m^{-2} s^{-1}), K is the equilibrium coefficient ($10^{-3.9993}$ at 100°C), and Q is the activity product, in this case equal to the concentration of $SiO_2(aq)$.

After 8 days, 1 kg of water is saturated with $SiO_2(aq)$ (Figure 8.3). The results from EQ3/6 and STORM are nearly identical.

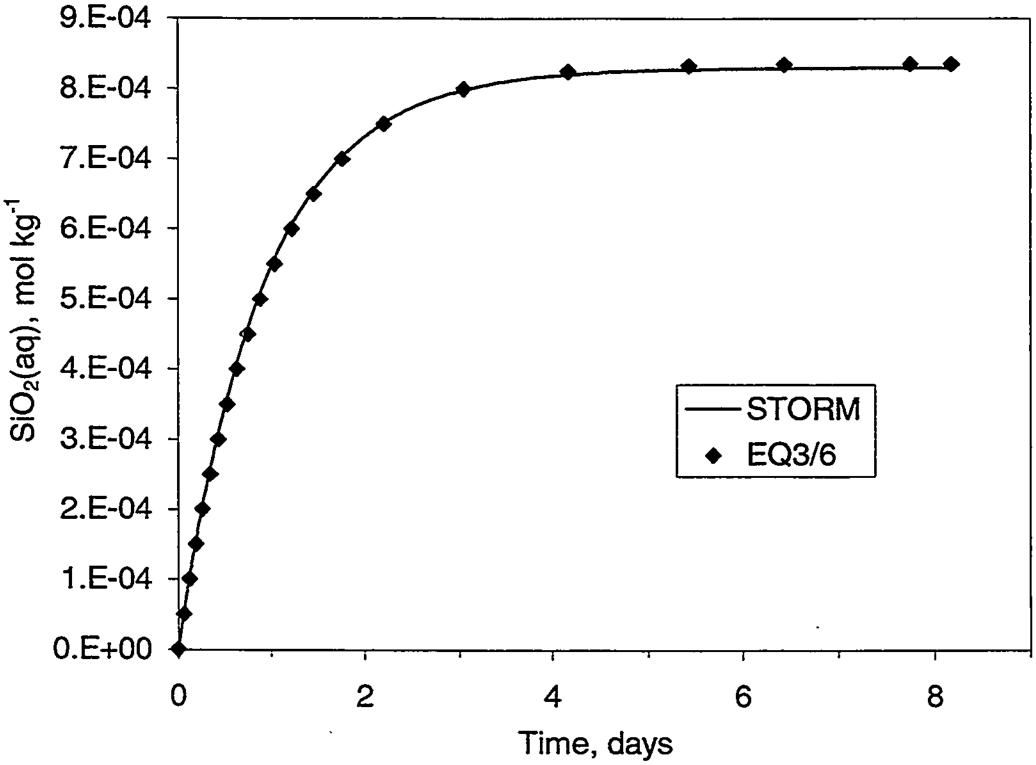

Figure 8.3. Aqueous Silica Concentration as Quartz Dissolves to Equilibrium in Deionized Water at 100°C

Data File 8.3. STORM Input File for Quartz Dissolution Test Case

```
~Simulation Title Card
1,
Glass PA 1-dimensional unsaturated flow,
DH Bacon,
Pacific Northwest National Laboratory,
July 1997,
00:00:00 AM PDT,
1,
Dissolve quartz for comparison to EQ3/6

~Solution Control Card
Normal,
Water w/ Transport,
1,
0.0,hr,8.179,day,1.,s,1.,hr,1.05,8,1.0e-6,
1.e+30,yr,1.e+30,yr,95001,
1,
Aqueous Relative Permeability,Geometric,
```

```
~Grid Card
Uniform Cartesian,
1,1,20,
1.0,m,
0.1,m,
0.1,m,

~Rock/Soil Zonation Card
1,
Soil,1,1,1,1,1,20,

~Mechanical Properties Card
Soil,2650.0,kg/m^3,0.5,0.5,1.e-5,1/m,Millington and Quirk,

~Hydraulic Properties Card
Soil,,,,,3.156e-5,hc cm/s,Constant,

~Saturation Function Card
Soil,Nonhysteretic van Genuchten,0.044,1/cm,2.2,0.08494,,

~Aqueous Relative Permeability Card
Soil,Tabular,33,
3.5026E-09,3.1593E-07,
0.01225919,6.7082E-07,
0.02977233,1.7099E-06,
0.04728546,3.8458E-06,
0.0647986,7.8611E-06,
0.08231173,1.4901E-05,
0.117338,4.5067E-05,
0.15236427,0.00011488,
0.18739054,0.00025837,
0.22241681,0.00052812,
0.25744308,0.00100111,
0.29246935,0.0017854,
0.32749562,0.00302769,
0.36252189,0.00492173,
0.39754816,0.00771754,
0.43257443,0.01173155,
0.4676007,0.01735761,
0.50262697,0.02507879,
0.53765324,0.03548017,
0.57267951,0.04926245,
0.60770578,0.06725641,
0.64273205,0.09043837,
0.67775832,0.11994642,
0.71278459,0.15709761,
0.74781086,0.20340604,
0.78283713,0.26060184,
0.8178634,0.330651,
0.85288967,0.41577624,
```

```
0.88791594,0.51847864,
0.92294221,0.64156028,
0.95796848,0.78814774,
0.99299475,0.96171657,
1,1,

~Gas Relative Permeability Card
Soil,Tabular,33,
1,1,
0.98774081,0.99999139,
0.97022767,0.99994918,
0.95271454,0.99987151,
0.9352014,0.99975797,
0.91768827,0.99960795,
0.882662,0.99919528,
0.84763573,0.99862545,
0.81260946,0.99788751,
0.77758319,0.99696733,
0.74255692,0.99584706,
0.70753065,0.99450437,
0.67250438,0.99291148,
0.63747811,0.99103385,
0.60245184,0.98882835,
0.56742557,0.98624081,
0.5323993,0.98320258,
0.49737303,0.97962579,
0.46234676,0.9753964,
0.42732049,0.97036419,
0.39229422,0.96432763,
0.35726795,0.95701052,
0.32224168,0.94802483,
0.28721541,0.93680881,
0.25218914,0.92251973,
0.21716287,0.90383668,
0.1821366,0.87857318,
0.14711033,0.84284783,
0.11208406,0.78910152,
0.07705779,0.70059708,
0.04203152,0.53266493,
0.00700525,0.13244732,
0,0,

~Initial Conditions Card
Gas Pressure,Aqueous Saturation,
3,
Gas Pressure,101325.0,Pa,0.0,1/m,0.0,1/m,0.0,1/m,1,1,1,1,1,20,
Aqueous Saturation,0.2,,0.0,1/cm,0.0,1/cm,0.0,1/cm,1,1,1,1,1,20,
Temperature,100.,C,,,,,,,1,1,1,1,1,20,

~Boundary Conditions Card
2,
```

```
Top,Neumann Aqueous,Initial Cond,Initial Cond,Initial Cond,Initial Cond,
1,1,1,1,20,20,1,
0.0,hr,-2.00E-10,m/s,,,,,,,,,,
Bottom,Free Gradient,Outflow,Outflow,Outflow,Outflow,
1,1,1,1,1,1,1,
0.0,hr,,,,,,,,,,,,,,

~Source Card
1,
Aqueous Mass,1,1,1,1,1,20,1,
0.,s,0.,kg/s,,

~Output Control Card
1,
1,1,19,
1,1,day,m,3,5,5,
1,
Solute Aqueous Conc,SiO2(aq),1/m^3,
9,
0.0,yr,
1.0,s,
1.0,min,
1.0,hr,
1.0,day,
2.0,day,
3.0,day,
4.0,day,
5.0,day,
11,
Gas Pressure,Pa,
Aqueous Pressure,Pa,
Aqueous Saturation,,
Rock/Soil Type,,
Diffusive Porosity,,
Aqueous Relative Permeability,,
z Aqueous Volumetric Flux,m/s,
Solute Aqueous Conc,H+,1/m^3,
Solute Aqueous Conc,H2O,1/m^3,
Solute Aqueous Conc,OH-,1/m^3,
Solute Aqueous Conc,SiO2(aq),1/m^3,

~Surface Flux Card
1,
Aqueous Volumetric Flux,m^3/s,m^3,Top,1,1,1,1,20,20,

~Reactive Transport Control card
Normal,
Fully Coupled,
0,0,0,0,1,0,0,0,0,
 2.00000E-06, 1.00000E+00, 1.00000E-06, 1.00000E-07, 1.00000E-03, 5.00000E-03,
2.00000E-03,
```

```
~Solute/Fluid Interaction Card
4,
H+,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
H2O,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
OH-,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
SiO2(aq),Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
0,

~Solute/Porous Media Interaction Card
soil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
SiO2(aq),0.0,m^3/kg,

~Aqueous Species card
   4,
'H+           ', 1.0000, 9.0000, 1,
 1.0000,' H',
'H2O          ', 0.0000,-4.0000, 2,
 2.0000,' H',
 1.0000,' O',
'OH-          ',-1.0000, 3.0000, 2,
 1.0000,' H',
 1.0000,' O',
'SiO2(aq)     ', 0.0000, 4.0000, 2,
 1.0000,'Si',
 2.0000,' O',

~Gas Species card
   0,

~Components card
   1,
'Si(total)    ',28.0855,  1,
 1.0000,'SiO2(aq)     ',

~Solid Species card
   1,
'Quartz       ', 2.6480,  2,
 1.0000,'Si',
 2.0000,' O',

~Fast Reactions card
   1,
   3,
-1.0000,'H2O          ',
 1.0000,'H+           ',
 1.0000,'OH-          ',
   5,    1.00000,
```

```
-1.49400E+01, 4.17100E-02,-1.91700E-04, 4.95000E-07,-6.15900E-10,

~Kinetic Aqueous Reactions card
   0,

~Slow Reactions card
    1,
  1,  1,
-1.0000,'Quartz      ',
   1,
 1.00000E+00,'SiO2(aq)    ', 1.0000, 1.0000,
-2.46350e+01,  0.,    1.00000, 0., 1,
    -3.0782,

~Fluid Decay card
   0,

~Solid Decay card
   0,

~Lithology card
'soil        ',  1,
'Quartz      ', 1.00000E+00, 0.60000E-05,

~Reaction BC card
   1,
1,1,
   0,
   0,
   0,
   3,
  1.00000E+00,  1,
 1.0000,'H2O          ',
  1.00000E-07,  1,
 1.0000,'H+           ',
  1.00000E-12,  1,
 1.0000,'SiO2(aq)     ',

~Reaction IC card
   1,
```

8.5 Calcite Precipitation

As an example of a slightly more complicated kinetic reaction path, calcite precipitates from supersaturated solution at 25°C according to the reaction

$$\mathit{Calcite} + \mathrm{H}^+ \leftrightarrow \mathrm{Ca}^{2+} + \mathrm{HCO}_3^- \tag{8.11}$$

Calcite dissolves according to the rate law (Reddy et al. 1981)

$$r_{calcite} = A_s k\left(1-\frac{Q}{K}\right) \tag{8.12}$$

where $r_{calcite}$ is the dissolution rate (mol s^{-1}), A_s is the surface area of the mineral (0.24168 m^2), k is the reaction rate constant (7×10^{-7} mol m^{-2} s^{-1}), K is the equilibrium coefficient ($10^{1.8487}$ at 25°C), and Q is the ion-activity product, in this case equal to

$$Q = \frac{[Ca^{2+}][HCO_3^-]}{[H^+]} \tag{8.13}$$

Several other species participate in equilibrium reactions (Table 8.4) and are included in the simulations. The $CO_2(g)$ partial pressure is fixed at a partial pressure of $10^{-1.5421}$ bars. The initial total Ca concentration is 1.2732×10^{-2} mol kg^{-1} and the initial pH is 7.5815.

Precipitation of calcite in 1 kg of water proceeds rapidly, and virtually ceases after the solution has reached saturation (0.12 days). Results from EQ3/6 and STORM are nearly identical (Figure 8.4).

Table 8.4. Aqueous Species, Equilibrium Reactions, and Equilibrium Constants Related to Calcite Dissolution

Reaction	Log K At 25°C
$OH^- + H^+ \leftrightarrow H_2O$	13.9951
$CO_2(aq) + H_2O \leftrightarrow H^+ + HCO_3^-$	-6.3447
$CO_2(g) \leftrightarrow CO_2(aq)$	-1.4689
$CaHCO_3^+ \leftrightarrow Ca^{2+} + HCO_3^-$	-1.0467
$CaCO_3(aq) + H^+ \leftrightarrow Ca^{2+} + HCO_3^-$	7.0017
$CO_3^{2-} + H^+ \leftrightarrow HCO_3^-$	10.3288
$CaOH^+ + H^+ \leftrightarrow Ca^{2+} + H_2O$	12.8500

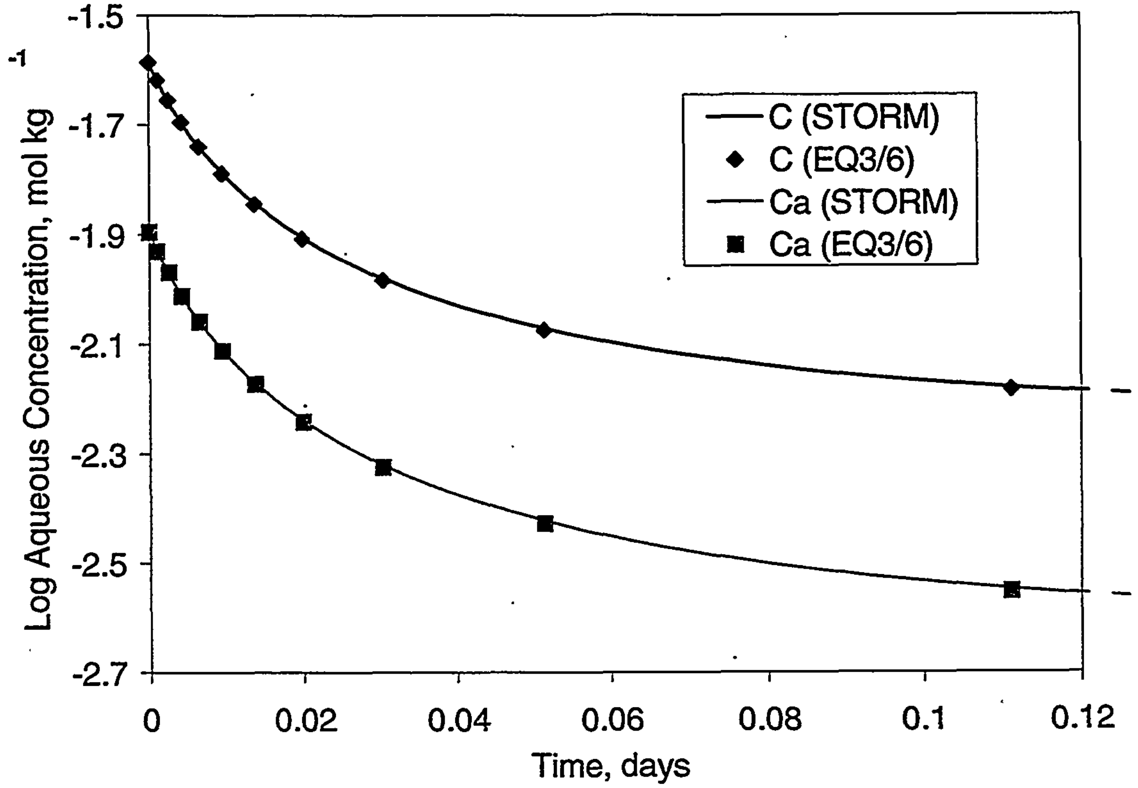

Figure 8.4. Total C and Ca Aqueous Concentrations as Calcite Precipitates from Supersaturated Solution

Data File 8.4. STORM Input File for Calcite Precipitation Test Case

```
~Simulation Title Card
1,
Glass PA 1-dimensional unsaturated flow,
DH Bacon,
Pacific Northwest National Laboratory,
July 1997,
00:00:00 AM PDT,
1,
Precipitate Calcite for comparison to EQ3/6

~Solution Control Card
Normal,
Water w/ Transport,
1,
0.0,hr,0.13,day,0.1,s,10.,day,1.05,8,1.0e-6,
1.e+30,yr,1.e+30,yr,95001,
1,
Aqueous Relative Permeability,Geometric,
```

```
~Grid Card
Uniform Cartesian,
1,1,20,
1.0,m,
0.1,m,
0.1,m,

~Rock/Soil Zonation Card
1,
Soil,1,1,1,1,1,20,

~Mechanical Properties Card
Soil,2650.0,kg/m^3,0.5,0.5,1.e-5,1/m,Millington and Quirk,

~Hydraulic Properties Card
Soil,,,,,3.156e-5,hc cm/s,Constant,

~Saturation Function Card
Soil,Nonhysteretic van Genuchten,0.044,1/cm,2.2,0.08494,,

~Aqueous Relative Permeability Card
Soil,Tabular,33,
3.5026E-09,3.1593E-07,
0.01225919,6.7082E-07,
0.02977233,1.7099E-06,
0.04728546,3.8458E-06,
0.0647986,7.8611E-06,
0.08231173,1.4901E-05,
0.117338,4.5067E-05,
0.15236427,0.00011488,
0.18739054,0.00025837,
0.22241681,0.00052812,
0.25744308,0.00100111,
0.29246935,0.0017854,
0.32749562,0.00302769,
0.36252189,0.00492173,
0.39754816,0.00771754,
0.43257443,0.01173155,
0.4676007,0.01735761,
0.50262697,0.02507879,
0.53765324,0.03548017,
0.57267951,0.04926245,
0.60770578,0.06725641,
0.64273205,0.09043837,
0.67775832,0.11994642,
0.71278459,0.15709761,
0.74781086,0.20340604,
0.78283713,0.26060184,
0.8178634,0.330651,
0.85288967,0.41577624,
0.88791594,0.51847864,
```

```
0.92294221,0.64156028,
0.95796848,0.78814774,
0.99299475,0.96171657,
1,1,

~Gas Relative Permeability Card
Soil,Tabular,33,
1,1,
0.98774081,0.99999139,
0.97022767,0.99994918,
0.95271454,0.99987151,
0.9352014,0.99975797,
0.91768827,0.99960795,
0.882662,0.99919528,
0.84763573,0.99862545,
0.81260946,0.99788751,
0.77758319,0.99696733,
0.74255692,0.99584706,
0.70753065,0.99450437,
0.67250438,0.99291148,
0.63747811,0.99103385,
0.60245184,0.98882835,
0.56742557,0.98624081,
0.5323993,0.98320258,
0.49737303,0.97962579,
0.46234676,0.9753964,
0.42732049,0.97036419,
0.39229422,0.96432763,
0.35726795,0.95701052,
0.32224168,0.94802483,
0.28721541,0.93680881,
0.25218914,0.92251973,
0.21716287,0.90383668,
0.1821366,0.87857318,
0.14711033,0.84284783,
0.11208406,0.78910152,
0.07705779,0.70059708,
0.04203152,0.53266493,
0.00700525,0.13244732,
0,0,

~Initial Conditions Card
Gas Pressure,Aqueous Saturation,
3,
Gas Pressure,101325.0,Pa,0.0,1/m,0.0,1/m,0.0,1/m,1,1,1,1,1,20,
Aqueous Saturation,0.2,,0.0,1/cm,0.0,1/cm,0.0,1/cm,1,1,1,1,1,20,
Temperature,25.,C,,,,,,,1,1,1,1,1,20,

~Boundary Conditions Card
2,
```

```
Top,Neumann Aqueous,Initial Cond,Initial Cond,Initial Cond,Initial Cond,Initial
Cond,Initial Cond,Initial Cond,Initial Cond,Initial Cond,Initial Cond,
1,1,1,1,20,20,1,
0.0,hr,-2.00E-10,m/s,,,,,,,,,,,,,,,,,,,,,,
Bottom,Free
Gradient,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,Outflow,Outfl
ow,
1,1,1,1,1,1,1,
0.0,hr,,,,,,,,,,,,,,,,,,,,,,,,,,,

~Source Card
1,
Aqueous Mass,1,1,1,1,1,20,1,
0.,s,0.,kg/s,,

~Output Control Card
1,
1,1,19,
1,1,day,m,3,5,5,
8,
Solute Aqueous Conc,H+,1/m^3,
Solute Aqueous Conc,CO2(aq),1/m^3,
Solute Aqueous Conc,HCO3-,1/m^3,
Solute Aqueous Conc,CO3--,1/m^3,
Solute Aqueous Conc,Ca++,1/m^3,
Solute Aqueous Conc,CaHCO3+,1/m^3,
Solute Aqueous Conc,CaCO3(aq),1/m^3,
Solute Aqueous Conc,CaOH+,1/m^3,
1,
0.1,s,
3,
Aqueous Saturation,,
Z Aqueous Volumetric Flux,cm/day,
Total Water Mass,kg,


~Surface Flux Card
1,
Aqueous Volumetric Flux,m^3/s,m^3,Top,1,1,1,1,20,20,

~Reactive Transport Control card
Normal,
Fully Coupled,
0,0,0,0,1,0,0,0,0,0,
 1.00000E-07, 1.00000E+00, 1.00000E-07, 1.00000E-07, 1.00000E-07, 1.00000E-07,
1.00000E-07,

~Solute/Fluid Interaction Card
10,
H+,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
H2O,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
```

```
OH-,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
HCO3-,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
Ca++,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
CO2(aq),Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
CaHCO3+,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
CaCO3(aq),Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
CO3--,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
CaOH+,Constant,5.0e-9,m^2/s,Continuous,1.e20,yr,
0,

~Solute/Porous Media Interaction Card
soil,0.,m,0.,m,
H+,0.0,m^3/kg,
H2O,0.0,m^3/kg,
OH-,0.0,m^3/kg,
HCO3-,0.0,m^3/kg,
Ca++,0.0,m^3/kg,
CO2(aq),0.0,m^3/kg,
CaHCO3+,0.0,m^3/kg,
CaCO3(aq),0.0,m^3/kg,
CO3--,0.0,m^3/kg,
CaOH+,0.0,m^3/kg,

~Aqueous Species Card
  10,
'H+          ', 1.0000, 9.0000, 1,
 1.0000,' H',
'H2O         ', 0.0000,-4.0000, 2,
 2.0000,' H',
 1.0000,' O',
'OH-         ',-1.0000, 3.0000, 2,
 1.0000,' H',
 1.0000,' O',
'HCO3-       ',-1.0000, 4.0000, 3,
 1.0000,' H',
 1.0000,' C',
 3.0000,' O',
'Ca++        ', 2.0000, 6.0000, 1,
 1.0000,'Ca',
'CO2(aq)     ', 0.0000, 3.0000, 2,
 1.0000,' C',
 2.0000,' O',
'CaHCO3+     ', 1.0000, 4.0000, 4,
 1.0000,'Ca',
 1.0000,' H',
 1.0000,' C',
 3.0000,' O',
'CaCO3(aq)   ', 0.0000, 0.0000, 3,
 1.0000,'Ca',
 1.0000,' C',
 3.0000,' O',
```

```
'CO3--        ', 2.0000, 5.0000, 2,
 1.0000,' C',
 3.0000,' O',
'CaOH+        ', 1.0000, 4.0000, 3,
 1.0000,'Ca',
 1.0000,' O',
 1.0000,' H',

~Gas Species card
   1,
'CO2(g)       ', 'CO2(aq)      ', 'fixed', 1.69443E-06, 2,
 1.0000,' C',
 2.0000,' O',
 1,
-1.4689,

~Components card
   2,
'CO2(total)  ',44.0100,  5,
 1.0000,'CO2(aq)      ',
 1.0000,'HCO3-        ',
 1.0000,'CO3--        ',
 1.0000,'CaHCO3+      ',
 1.0000,'CaCO3(aq)    ',
'Ca(total)   ',40.07800, 4,
 1.0000,'Ca++         ',
 1.0000,'CaHCO3+      ',
 1.0000,'CaCO3(aq)    ',
 1.0000,'CaOH+        ',

~Solid Species card
   1,
'Calcite', 2.7099,  3,
 1.0000,'Ca',
 1.0000,' C',
 3.0000,' O',

~Fast Reactions card
   6,
   2,
-1.0000,'OH-          ',
-1.0000,'H+           ',
   1,    1.00000,
13.9951,
   3,
-1.0000,'CO2(aq)      ',
 1.0000,'H+           ',
 1.0000,'HCO3-        ',
   1,    1.00000,
-6.3447,
   3,
```

```
-1.0000,'CaHCO3+     ',
 1.0000,'Ca++        ',
 1.0000,'HCO3-       ',
   1,    1.00000,
-1.0467,
   4,
-1.0000,'CaCO3(aq)   ',
-1.0000,'H+          ',
 1.0000,'Ca++        ',
 1.0000,'HCO3-       ',
   1,    1.00000,
7.0017,
   3,
-1.0000,'CO3--       ',
-1.0000,'H+          ',
 1.0000,'HCO3-       ',
   1,    1.00000,
10.3288,
   3,
-1.0000,'CaOH+       ',
-1.0000,'H+          ',
 1.0000,'Ca++        ',
   1,    1.00000,
12.8500,

~Kinetic Aqueous Reactions card
   0,

~Slow Reactions card
    1,
 -1, 1,
-1.0000,'Calcite',
   3,
-1.0000,'H+     ', -1.0000, -1.0000,
 1.0000,'Ca++   ',  1.0000,  1.0000,
 1.0000,'HCO3-  ',  1.0000,  1.0000,
-14.172,       0.,  1.0000,  0.0000, 1,
 1.8487,

~Fluid Decay card
   0,

~Solid Decay card
   0,

~Lithology card
'soil         ',  1,
'Calcite      ', 1.00000E+00, 1.24131E-02,

~Reaction BC card
   1,
```

```
1,1,
   0,
   0,
   1,
'CO2(g)      ', 2.8701E-02,
   3,
  1.0000E+00,  1,
 1.0000,'H2O          ',
  2.6209E-08,  1,
 1.0000,'H+           ',
  1.2732E-02,  4,
 1.0000,'Ca++         ',
 1.0000,'CaHCO3+      ',
 1.0000,'CaCO3(aq)    ',
 1.0000,'CaOH+        ',

~Reaction IC card
   1,
```

9.0 Example Simulations

Three simulations are presented to demonstrate unique features of STORM. The first simulation presents a 2-dimensional, waste-form-alone-design simulation of steady-state unsaturated flow and transient reactive transport of aqueous species released as the glass waste dissolves. In the first simulation, glass dissolves according to a rate law dependent on aqueous phase silica concentration and pH. The approach is new in that a full set of recent experimentally derived rate parameters for LD-5412 glass is used. The second simulation demonstrates how exchange of hydrogen for sodium in the glass enhances the rate of glass dissolution. This simulation approach is also new in that mass balance is preserved by tracking the masses of altered and unaltered glass separately. The third simulation demonstrates how gas phase diffusion of CO_2 may increase the dissolution rate of the glass. Transient gas phase transport is a new feature added to the code this fiscal year.

9.1 Base Case

The disposal facility is modeled as a 2-dimensional domain where the waste form, a borosilicate waste glass (LD6-5412), is surrounded by backfill soil above several layers of Hanford sediments and the water table (Figure 9.1). Laboratory-measured hydraulic properties are used for each porous material (Rockhold et al. 1993); the glass waste is assumed to have hydraulic properties equivalent to that of a gravel (Fayer et al. 1997). These properties are used to calculate the water saturation distribution below a surface barrier designed to allow a water flux of no more than 0.5 mm yr^{-1}. The coarse glass acts as a hydraulic barrier, causing water to pond on top of the waste form and flow around the sides (Figure 9.2).

Included in the reactive transport portion of the simulation were 22 aqueous species, 1 gaseous species, 9 minerals, 11 equilibrium reactions, and 9 kinetic reactions (Table 9.1).

The corrosion of silicate glasses in water is represented by a special type of irreversible dissolution reaction. A conventional transition state kinetic rate equation (Aagaard and Helgeson 1982) is used to develop a constitutive relationship to compute the flux of any element i released from the glass into the aqueous phase. Currently, H^+ is the only aqueous species shown to directly influence the dissolution rate, that gives

$$J_i^a = \upsilon_i \vec{k} e^{-\frac{E_a}{RT}} a_{H^+}^{-\eta} \left[1 - \left(\frac{Q}{K} \right)^{\sigma} \right], i = 1, 2, \ldots, N \tag{9.1}$$

where J_i^a is the flux of element i (mol m^{-2} s^{-1}), υ_i is the stoichiometric coefficient of element i in the glass, $\vec{k}$ is the intrinsic rate constant (mol m^{-2} s^{-1}), E_a is the activation energy (J mol^{-1}), R is the gas constant (J mol^{-1} K^{-1}), T is the temperature (K), a_{H^+} is the hydrogen ion activity, η is the order of the hydrogen ion activity dependence, Q is the ion-activity product of the rate controlling phase, K is the

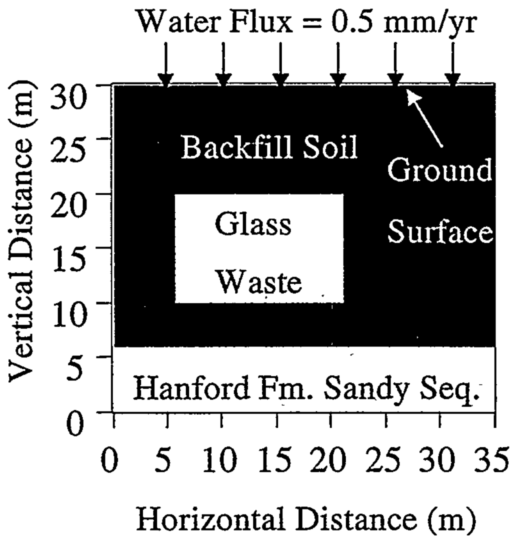

Figure 9.1. Sediment Types and Flow Boundary Condition Used in the Simulations

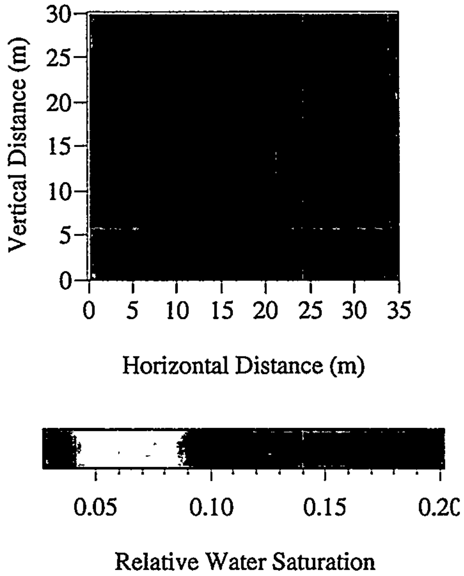

Figure 9.2. Steady State Unsaturated Flow Field

Table 9.1. List of Reactions

	(Note: 1-11 are equilibrium reactions, 12-21 are kinetic reactions)
1.	$H_2O = H^+ + OH^-$
2.	$CO_2(g) = CO_2(aq)$
3.	$CO_2(aq) + H_2O = H^+ + HCO_3^-$
4.	$HCO_3^- = H^+ + CO_3^{2-}$
5.	$Al(OH)_4^- + H^+ = Al(OH)_3 + H_2O$
6.	$H_2SiO_4^{2-} + 2H^+ = SiO_2(aq) + 2H_2O$
7.	$H_2SiO_3^- + H^+ = SiO_2(aq) + 2H_2O$
8.	$BO_2^- + H^+ + H_2O = B(OH)_3(aq)$
9.	$CaCO_3(aq) + H^+ = Ca^{2+} + HCO_3^-$
10.	$CaHCO_3^+ = Ca^{2+} + HCO_3^-$
11.	$CaOH^+ + H^+ = Ca^{2+} + H_2O$
12.	$Glass + 0.5OH^- + 0.45H_2O = 0.0202K^+ + 0.4208Na^+ 0.046Ca^{2+} + 0.094BO_2^- + 0.154Al(OH)_3(aq) + 0.2712HSiO_3^- + 0.3348H_2SiO_4^{2-} + 1.6172x10^{-6}TcO_4^{2-}$
13.	$Quartz + OH^- = HSiO_3^-$
14.	$Calcite = Ca^{2+} + CO_3^{2-}$
15.	$Albite(high) + 2OH^- = Na^+ + Al(OH)_3(aq) + 3HSiO_3^-$
16.	$K-feldspar + 2OH^- = K^+ + Al(OH)_3(aq) + 3HSiO_3^-$
17.	$Illite + 2.4OH^- = 0.6K^+ + 0.25Mg^{2+} + 2.3Al(OH)_3(aq) + HSiO_3^-$
18.	$Analcime + 0.96H^+ = 0.96Na^+ + 0.96Al(OH)_3(aq) + 2.04SiO_2(aq) + 5.04H_2O$
19.	$Chalcedony = SiO_2(aq)$
20.	$NaAlSi_2O_6 \cdot 6H_2O + OH^- = Na^+ + Al(OH)_3(aq) + 2HSiO_3^- + 4H_2O$

equilibrium constant of the rate controlling phase, and σ is the net reaction order. Assigning K to a simple SiO_2 polymorph, such as chalcedony, yields the best agreement with experimental data for LD6-5412 glass (McGrail et al. 1997). Consequently, K, the equilibrium constant for chalcedony, is assigned for the simulations discussed here. Experiments utilizing a single-pass flow-through apparatus for measuring glass dissolution rates at various pH and temperatures yield values for the parameters in Equation (2.16), as shown in Table 9.2.

Table 9.2. Glass Dissolution Rate Parameters

Parameter	Value
$\bar{k}$	$10^{2.05}$ g m^{-2} s^{-1}
η	0.40
E_a	74.8 kJ mol^{-1}
σ	1

The subsurface is assumed open to the atmosphere, and the subsurface gas phase concentrations are not limited by diffusion, so that $CO_2(g)$ concentrations are a constant 0.035 percent by volume. The diffusion coefficient for all species in pure water is assumed to be 5×10^{-9} m^2 s^{-1}. As time advances, the pH rises to a maximum value of 8.1 because dissolution of the glass releases alkali metals (Na, K) (Figure 9.3). The largest glass dissolution rates (6.3×10^{-11} mol s^{-1}) appear at the interface between the vault and the surrounding soil because the steep concentration gradient into the soil lowers the concentration of Si near the interface (Figure 9.4). Concentrations of ^{99}Tc are smooth because under the oxidizing conditions in the disposal facility, ^{99}Tc is highly soluble as a pertechnitate ($^{99}TcO_4^-$) anion and so is subject to aqueous phase diffusion (Figure 9.5). A maximum $^{99}TcO_4^-$ concentration of 1.9×10^{-9} mol kg^{-1} is observed at the bottom of the waste form. Concentrations of ^{99}Tc are presented because this species provides the dominant contribution to the drinking water dose (Mann et al. 1998).

9.2 Ion Exchange

Experimental results (McGrail et al. 1997) suggest the exchange of sodium for hydrogen in the glass waste form

$$\text{Glass}\cdots0.4208\text{Na} + 0.4208\text{H}^+ \rightleftharpoons \text{Glass}\cdots0.4208\text{H} + 0.4208\text{Na}^+ \tag{9.2}$$

is important at the low temperatures expected in the waste form. When this reaction is included, the maximum pH in the waste form increases by one pH unit to a value of 9.1 (Figure 9.6). As would be predicted by Equation (2.15), the higher pH causes the glass dissolution rate to increase to a maximum value of 7.4×10^{-11} mol s^{-1} at the edges of the waste form (Figure 9.7). The increase in glass dissolution results in the maximum Tc concentration (Figure 9.8) also increasing to a value of 3.0×10^{-9} mol kg^{-1}. The increases in pH and Tc concentrations are significantly less than those predicted in previous analyses (Mann et al. 1998), due to the buffering capacity provided by assuming a fixed partial pressure of $CO_2(g)$.

9.3 Gas Diffusion

In the previous two simulations, the assumption is that partial pressure of CO_2 in the gas phase was constant, implying the diffusion rate of $CO_2(g)$ through the vadose zone

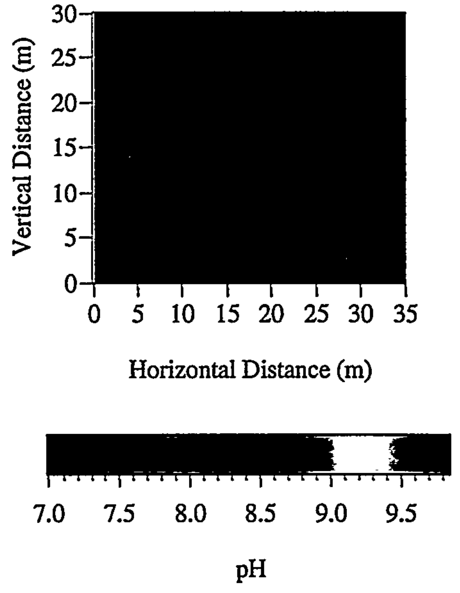

Figure 9.3. pH After 20,000 Years, No Ion Exchange, Fixed P_{CO_2}

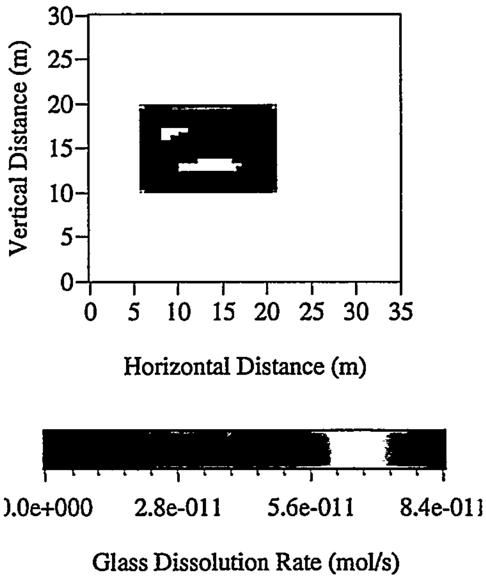

Figure 9.4. Glass Dissolution Rate After 20,000 Years, No Ion Exchange, Fixed P_{CO_2}

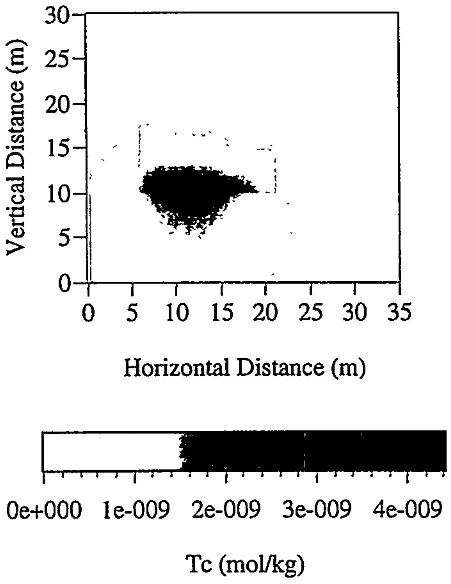

Figure 9.5. Tc Concentration After 20,000 Years, No Ion Exchange, Fixed P_{CO_2}

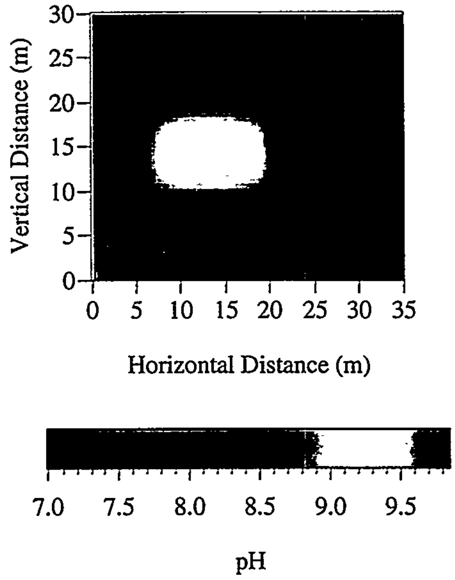

Figure 9.6. pH After 20,000 Years, with Ion Exchange, Fixed P_{CO_2}

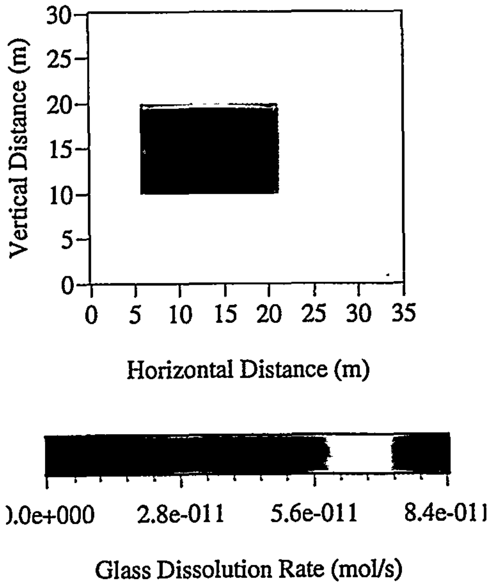

Figure 9.7. Glass Dissolution Rate After 20,000 Years, with Ion Exchange, Fixed P_{CO_2}

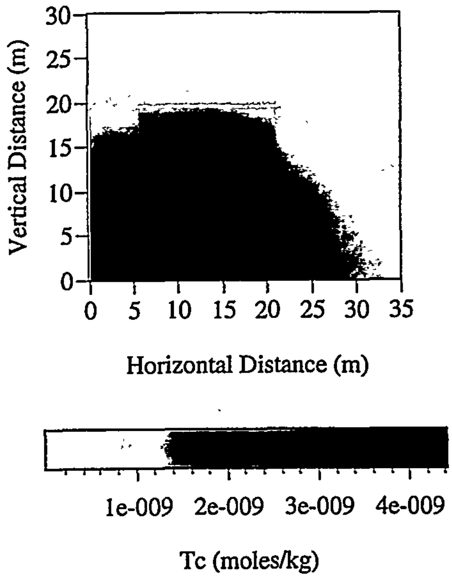

Figure 9.8. Tc Concentration After 20,000 Years, with Ion Exchange, Fixed P_{CO_2}

is fast relative to the rate of consumption of CO_2(aq) in aqueous complexation and dissolution-precipitation reactions. This assumption is relaxed by assuming a diffusion coefficient for CO_2(g) of 1.6×10^{-6} m^2 s^{-1} and allowing STORM to transport CO_2(g) throughout the spatial domain. The results (Figure 9.9) show that CO_2(g) concentrations in the center of the vault decrease to a partial pressure nearly an order of magnitude below atmospheric pressure. This reduces the concentration of carbonic acid produced from dissolution of CO_2(g) into the pore water, thus causing the pH to increase to a maximum value of 9.8 (Figure 9.10). Again, the increase in pH causes the glass dissolution rate (Figure 9.11), both on the edges and at the center of the waste form, and the aqueous Tc concentrations (Figure 9.12) to increase accordingly.

9.4 Conclusions

These three simulations demonstrate the long-term performance of the low-activity waste glasses can be successfully simulated with the STORM code under realistic field conditions of 2-phase flow of water and air. The results show that ion exchange and gas phase diffusion can have a significant effect on radionuclide release rates from the glass waste form. Assuming equilibrium with atmospheric CO_2(g) throughout the spatial domain does not appear to be appropriate and can lead to non-conservative radionuclide release rates from the disposal system.

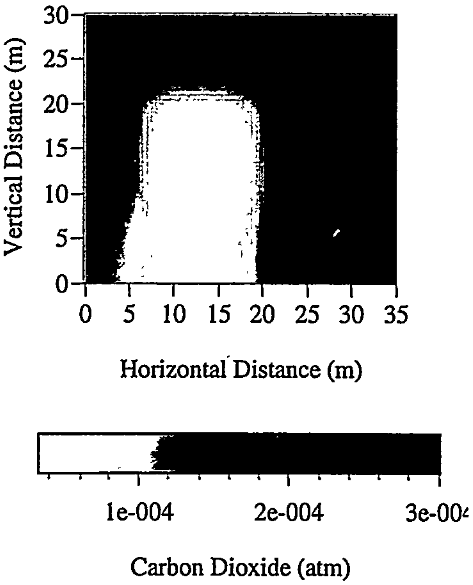

Figure 9.9. Gas Phase CO_2 Concentrations After 20,000 Years, with Ion Exchange and CO_2 Diffusion

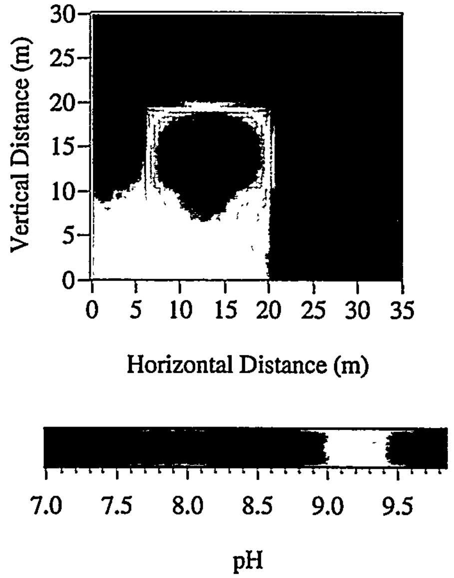

Figure 9.10. pH After 20,000 Years, with Ion Exchange and CO_2 Diffusion

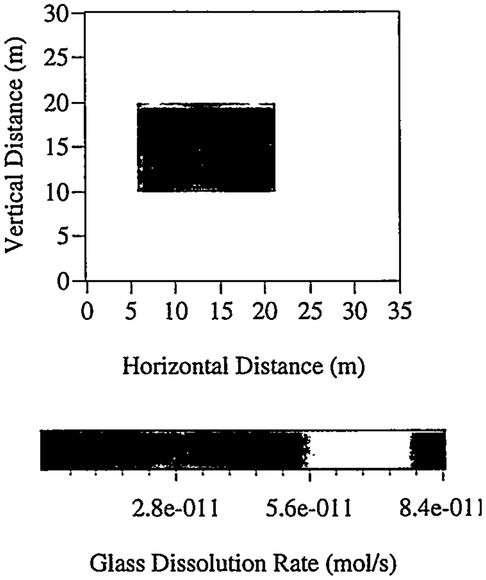

Figure 9.11. Glass Dissolution Rate After 20,000 Years, with Ion Exchange and CO_2 Diffusion

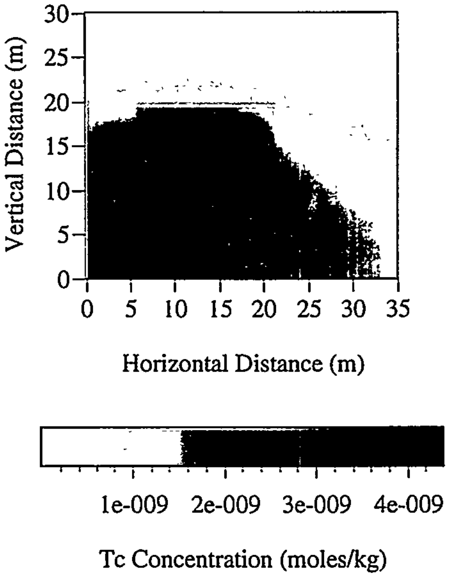

Figure 9.12. Tc Concentrations After 20,000 Years, with Ion Exchange and CO_2 Diffusion

10.0 References

Aagaard P, and HC Helgeson. 1982. *Am. J. Sci.*, 282:237-.

Barry DA, CT Miller, and PJ Culligan-Hensley. 1996. "Temporal discretisation errors in non-iterative split-operator approaches to solving chemical reaction/groundwater transport models." *J. Contam. Hydrol.*, 22:1-17.

Carman PC. 1937. "Fluid flow through a granular bed." *Trans. Instn. Chem. Engrs. (London)*, 15:150-156.

Chen Y, DW Engel, BP McGrail, and KS Lessor. 1995. *AREST-CT V1.0 Software Verification.* PNL-10692, Pacific Northwest Laboratory, Richland, Washington.

Chen Y, BP McGrail, and DW Engel. 1997. "Source-term analysis for Hanford low-activity tank waste using the reaction-transport code AREST-CT." *Scientific Basis for Nuclear Waste Management XX*, Pittsburgh.

Chen Y, P McGrail, and DW Engel. 1996. "A reaction transport model and its application to performance assessment of nuclear waste disposal." *International Conference on Deep Geological Disposal of Radioactive Waste, Canadian Nuclear Society.*

Davies CW. 1962. *Ion Association.* Butterworths, London.

Dongarra JJ, JR Bunch, CB Moler, and GW Stewart. 1979. *LINPACK User's Guide.* Society of Industrial and Applied Mathematicians (SIAM), Philadelphia, Pennsylvania.

Ecology. 1996. Hanford Facility Agreement and Consent Order, Sixth Amendment, , Washington State Department of Ecology, United States Environmental Protection Agency, and United States Department of Energy.

Fair GM, and LP Hatch. 1933. "Fundamental factors governing the streamline flow of water through sand." *J. Am. Wat. Wks. Ass.*, 25:1551-1565.

Fayer MJ, MD White, and CT Kincaid. 1997. *Sensitivity Tests of the Waste-Form-Alone Design for the Low-Activity-Waste Disposal System.* PNNL-11717, Pacific Northwest National Laboratory, Richland, Washington.

Golub GH, and CF Van Loan. 1983. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, Maryland.

Gupta AD, LW Lake, GA Pope, K Sepehrnoori, and MJ King. 1991. "High-resolution monotonic schemes for reservoir fluid flow simulation." *In Situ*, 15(3):289-317.

Harten A. 1983. "High resolution schemes for hyperbolic conservation laws." *J. Comp. Phys.*, 49:357.

Helgeson HC. 1969. "Thermodynamics of hydrothermal systems at elevated temperatures and pressures." *American Journal of Science*, 267:729-804.

Kirkner DJ, and H Reeves. 1988. "Multicomponent mass transport with homogeneous and heterogeneous chemical reactions: Effect of the chemistry on the choice of numerical algorithm 1. Theory." *Water Resour. Res.*, 24:1719-1729.

Kreyszig E. 1979. *Advanced Engineering Mathematics.* John Wiley & Sons, Inc., New York.

Lichtner PC. 1985. "Continuum model for simultaneous chemical reaction and mass transport in hydrothermal systems." *Geochim. Cosmochim. Acta*, 49:779-800.

Mangold DC, and C-F Tsang. 1991. "A summary of subsurface hydrological and hydrochemical models." *Rev. Geophys.*, 29:51-79.

Mann FM, RP Puigh, II, CR Eiholzer, Y Chen, NW Kline, AH Lu, BP McGrail, PD Rittmann, GF Williamson, NR Brown, and PE LaMont. 1998. *Hanford Immobilized Low Activity Tank Waste*

Performance Assessment. DOE/RL-97-69 Rev. B, Project Hanford Management Contractor, Richland, Washington.

McGrail BP, and DH Bacon. 1998. *Selection of a Computer Code for Hanford Low-Level Waste Engineered-System Performance Assessment.* PNNL-10830 Rev. 1, Pacific Northwest National Laboratory, Richland, Washington.

McGrail BP, WL Ebert, AJ Bakel, and DK Peeler. 1997. "Measurement of kinetic rate law parameters on a Na-Ca-Al borosilicate glass for low-activity waste." *J. Nuclear Materials*, 249:175-189.

Oppe TC, WD Joubert, and DR Kincaid. 1988. *NSPCG User's Guide, Version 1.0, A Package for Solving Large Sparse Linear Systems by Various Iterative Methods.* CNA-216, University of Texas, Austin, Texas.

Ortoleva P, E Merino, C Moore, and J Chadam. 1987. "Geochemical self-organization II: The reactive-infiltration instability." *Am. J. Sci.*, 287:1008-1040.

Patankar SV. 1980. *Numerical Heat Transfer and Fluid Flow.* Hemisphere Publishing Corporation, Washington, D.C.

Pitzer KS. 1973. "Thermodynamics of electrolytes - 1. Theoretical basis and general equations." *Journal of Physical Chemistry*, 4:249-265.

Press WH, SA Teukolsky, WT Vetterling, and BP Flannery. 1986. *Numerical Recipes in FORTRAN: The Art of Scientific Computing.* Cambridge University Press.

Reddy MM, LN Plummer, and E Busenberg. 1981. "Crystal growth of calcite from calcium bicarbonate solutions at constant PCO2 and 25C: A test of a calcite dissolution model." *Geochim. et Cosmochim. Acta*, 45:1281-1289.

Reed MH. 1982. "Calculation of multicomponent chemical equilibria and reaction processes in system involving minerals, gases and and aqueous phase." *Geochim. Cosmochim. Acta*, 46:513-528.

Rimstidt JD, and HL Barnes. 1980. "The kinetics of silica-water reactions." *Geochim. et Cosmochim. Acta*, 44:1683-1699.

Rockhold ML, MJ Fayer, and PR Heller. 1993. *Physical and Hydraulic Properties of Sediments and Engineered Materials Associated with Grouted Double-Shell Tank Waste Disposal at Hanford.* PNL-8813, Pacific Northwest National Laboratory, Richland, Washington.

Sevougian SD, RS Schechter, and LW Lake. 1993. "Effect of partial local equilibrium on the propagation of precipitation/dissolution waves." *Ind. Eng. Chem. Res.*, 32:2281-2304.

Steefel CI, and AC Lasaga. 1994. "A coupled model for transport of multiple chemical species and kinetic precipitation/dissolution reactions with application to reactive flow in single phase hydrothermal systems." *Am. J. Sci.*, 294:529-592.

Strikwerda JC. 1989. *Finite Difference Schemes and Partial Differential Equations.* Chapman & Hall, New York.

Sweby PK. 1984. "High resolution schemes using flux limiters for hyperbolic conservation laws." *SIAM J. Numer. Anal*, 21:995.

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

Valocchi AJ, and M Malmstead. 1992. "Accuracy of operator splitting for advection-dispersion-reaction proplems." *Water Resour. Res.*, 28:1471-1476.

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

Wolery TJ. 1992. *EQ3NR, A Computer Program for Geochemical Aqueous Speciation-Solubility Calculations: Theoretical Manual, User's Guide, and Related Documentation (Version 7.0)*. UCRL-MA-110662 PT III, Lawrence Livermore National Laboratory, Livermore, California.

Yeh GT, and VS Tripathi. 1989. "A critical evaluation of recent developments in hydrogeochemical transport models of reactive multichemical components." *Water Resour. Res.*, 25:93-108.

Yeh GT, and VS Tripathi. 1991. "A model for simulating transport of reactive multispecies components: model development and demonstration." *Water Resour. Res.*, 27:3075-3094.

Yeh T-CJ, R Srivastava, A Guszman, and T Harter. 1993. "A numerical model for water flow and chemical transport in variably saturated porous media." *Ground Water*, 31:634-644.

Appendix A

STORM Version 2 Input Data Format

Appendix A

STORM Version 2 Input Data Format

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.2 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.3 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 ] }
    { Initial Conditions }
Else:
    { Normal }
    { Restart }
    { Initial Conditions }
Endif:
Format: Char[a],
```

```
Operational Mode Option[a]
    { Water }
    { Water-Air }
    { Water-Air-Energy }
    Format: Char[a],
```

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

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

```
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:
```

```
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:

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 }
{ Gas Density } { Gas Relative Permeability }
{ Gas Viscosity } { Hydraulic Dispersion }
{ Intrinsic Permeability } { Thermal Conductivity }
{ 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.4 Solution Control Card Example

```
~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, Upwind,
Aqueous Relative Permeability, Upwind,
Intrinsic Permeability, Harmonic,
```

A.5 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], (note: cannot be greater than 1 in current version)
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
```

```
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.6 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.7 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.8 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.9 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 }
{ Free Gradient } { Hydraulic Gradient } { Initial Condition }
Format: *Char*^{a}, *Char*^{b}, <*Char*^{c},>

Elseif: Operational Mode Option: Solution Control Card = { Water-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 }
Format: *Char*^{a}, *Char*^{b}, *Char*^{c}, <*Char*^{d},>

Elseif: Operational Mode Option: Solution Control Card = { Water-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 }
Format: *Char*^{a}, *Char*^{b}, *Char*^{c}, *Char*^{d}, <*Char*^{e},>

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},

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 }
            Base Aqueous Pressure[c], Units[d] (Pa),
        Else: Null[c], Null[d],
        Endif:
        Format: Real[a], Char[b], Real[c], Char[d], Real[e], Char[f],
            Real[e], Char[f], <Real[e], Char[f],>
    Endfor: Number of Boundary Times
```

```
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[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:
        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]
        Format: Real[a], Char[b], Real[c], Char[d], Real[e], Char[f],
            Real[g], Char[h], Real[g], Char[h], <Real[g], Char[h],>
    Endfor: Number of Boundary Times
```

```
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:
        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]
        Format: Real[a], Char[b], Real[c], Char[d], Real[e], Char[f], Real[g], Char[h], Real[i],
            Real[j], Char[k], Real[j], Char[k], <Real[j], Char[k],>
    Endfor: Number of Boundary Times
Endfor: Number of Boundary Condition Domains

Endcard: Boundary Conditions Card
```

A.10 Boundary Conditions Card Examples

```
#-------------------------------------------------------
~Boundary Conditions Card
#-------------------------------------------------------
2,
East,Hydraulic Gradient,
50,50,1,1,1,113,1,
0,day,183254,Pa,
West,Neumann,Outflow,
3,3,1,1,14,33,6,
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,,,
149.835,min,-0.000203923,ft/sec,,,
187.29,min,-0.000206671,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,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,
1,1,1,1,1,10,1,
0,yr,6.6e-5,m/s,
East,Hydraulic Gradient,
20,20,1,1,1,10,1,
0,yr,121557.98,Pa,
```

```
~Boundary Conditions Card
2,
Top,Neumann,Zero Flux,
1,1,1,1,92,92,1,
0,hr,-3.32,cm/hr,,,
Bottom,Dirichlet,Dirichlet,
1,1,1,1,1,1,1,
0,hr,111119.948,Pa,111119.948,Pa,
```

A.11 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

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

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

Reference Node Variable and Plot File Variable Options

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

Reference Node Variable and Plot File Variable Options

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

Reference Node Variable and Plot File Variable Options

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

{ znc aqueous vol [umetric flux (node centered)] }
{ znc gas vol [umetric flux (node centered)] } { znc heat flux [(node centered)] }

Default Units for the Reference Node and Plot File Variables

Air Concentration (kg/m^3)
Water Concentration (kg/m^3)
Aqueous Relative Permeability (null)
Aqueous Volumetric Flux (m/s)
Energy Source Integral (J)
Gas Sat [uration] (null)
Gauge Pressure (Pa)
Heat Flux (W/m^2)
Phase Condition (null)
Relative Permeability (null)
Temperature (C)
Air Source Integral (kg)
Water Source Integral (kg)
Aqueous Sat [uration] (null)
Density (kg/m^3)
Gas Relative Permeability (null)
Gas Volumetric Flux (m/s)
Head (m),
Moisture Content (null)
Pressure (Pa)
Sat [uration] (null)
Thermal Conductivity (W/m K)

A.12 Output Control Card Example

```
~Output Control Card
4,
20,1,1,
20,1,3,
20,1,6,
20,1,10,
1,1,yr,m,,,,
3,
Aqueous Pressure,Pa,
Aqueous Saturation,,
X Aqueous Volumetric Flux,m/s,
2,
1.0, s,
100,yr,
3,
Aqueous Density,kg/m^3,
Aqueous Pressure,Pa,
Aqueous Sat,,
```

A.13 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 } { Aqueous Mass } { Well }
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 }
Source Type Option[a]
{ Aqueous Volumetric } { Aqueous Mass } { Gas Volumetric w/ Mass Fraction }
{ Gas Volumetric w/ Relative Humidity } { Gas Mass w/ Mass Fraction }
{ Gas Mass w/ Relative Humidity }
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 } { Power Density } { Aqueous Volumetric } { Aqueous Mass }
{ Gas Volumetric w/ Mass Fraction } { Gas Volumetric w/ Relative Humidity }
{ Gas Mass w/ Mass Fraction } { Gas Mass w/ Relative Humidity }
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],

```
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),
        Elseif: Source Type Option = { Aqueous Mass }
            Aqueous Mass Rate[c], Units[d] (kg/s),
        Elseif: Source Type Option = { Well }
            Aqueous Mass Rate[c], Units[d] (kg/s),
            Borehole Radius[e], Units[f] (m),
            Well Bottom Pressure Constraint[g], Units[h] (Pa),
        Format: Real[a], Char[b], Real[c], Char[d], [ Real[e], Char[f], Real[g], Real[h], Char[i], Real[j], ]
    Endfor: Number of Source Times
```

```
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 = { 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]
        Format: Real[a], Char[b], Real[c], Char[d], [ Real[e], Char[f], Real[g], ]
    Endfor: Number of Source Times
```

```
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 = { Gas Mass w/ Mass Fraction }
            Temperature[c], Units[d] (C), Pressure[e], Units[f] (Pa),
            Gas Mass 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 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 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]
        Format: Real[a], Char[b], Real[c], Char[d], [ Real[e], Char[f], Real[g], Char[h], Real[i], ]
    Endfor: Number of Source Times
```

A.14 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,
```

A.15 Saturation Function Card

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

For: Number of Rock/Soil Types

```
If: Operational Mode Option: Solution Control Card = { Water } { Water-Air } or { Water-Air-Energy }
        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],
```

```
          Entry Head Drainage[c], Units[d] (m),  Entry Head Imbibition[e], Units[f] (m),
      Elseif:  Saturation Function Option = { Hysteretic Brooks and Corey }
          “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],
  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:
```

A.16 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,Nonhysteretic Brooks and Corey,10.0,cm,0.383,,
SM-SP1,Nonhysteretic 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.17 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 }
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],*

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]
{ Aqueous Pressure } { Gas Pressure } { Temperature } { Aqueous Saturation }

Elseif: Operational Mode Option: Solution Control Card = { Water-Air }
or { Water-Air-Energy }
Variable Name Option[a]
{ Aqueous Pressure } { Gas Pressure } { Temperature }
{ Aqueous Saturation } { Air Partial Pressure } { Air Gas Mole Fraction }

{ Air Aqueous Mole Fraction } { Trapped Air Saturation }

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],

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 }
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],

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],

Elseif: Variable Name Option = { Relative Humidity }
Relative Humidity[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],

Endfor: Number of Initial Conditions Domains

Endcard: Initial Conditions Card

A.18 Initial Conditions Card Examples

```
~Initial Conditions Card
Aqueous Pressure,Gas Pressure,
4,
```

```
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,
```

```
~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,
```

A.19 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 }
- 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-Air } { Water-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],
            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:
Endfor: Number of Rock/Soil Types

Endcard: Mechanical Properties Card
```

A.20 Mechancial 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.21 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. Matrix Intrinsic Perm. or Hydraulic Cond.[b], Units[c] (m^2 or hc m/s),
Y-Dir. Matrix Intrinsic Perm. or Hydraulic Cond.[d], Units[e] (m^2 or hc m/s),
Z-Dir. Matrix Intrinsic Perm. or Hydraulic Cond.[f], Units[g] (m^2 or hc m/s),
X-Dir. Fracture Intrinsic Perm. or Hydraulic Cond.[h], Units[i] (m^2 or hc m/s),
Y-Dir. Fracture Intrinsic Perm. or Hydraulic Cond.[j], Units[k] (m^2 or hc m/s),
Z-Dir. Fracture Intrinsic Perm. or Hydraulic Cond.[l], Units[m] (m^2 or hc m/s),
Intrinsic Permeability Function[n],
{Constant } {Kozeny and Carmen } { Fair and Hatch }
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], *Char*[n],
Else:
X-Dir. Intrinsic Permeability or Hydraulic Conductivity[b], Units[c] (m^2 or hc m/s),
Y-Dir. Intrinsic Permeability or Hydraulic Conductivity[d], Units[e] (m^2 or hc m/s),
Z-Dir. Intrinsic Permeability or Hydraulic Conductivity[f], Units[g] (m^2 or hc m/s),

Intrinsic Permeability Function[h],
{Constant } {Kozeny and Carmen } { Fair and Hatch }
Format: *Char*[a], *Real*[b], *Char*[c], *Real*[d], *Char*[e], *Real*[f], *Char*[g], *Char*[h],
Endif:
Endfor: Number of Rock/Soil Types

Endcard: Hydraulic Properties Card

A.22 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,Fair and Hatch,
```

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

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

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

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

A.23 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.24 Thermal Properties Card Examples

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

A.25 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.26 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,
```

```
~Rock/Soil Zonation Card
1,
Sand1,1,20,1,1,1,10,
```

A.27 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 }
    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$^d$,
    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
    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
    Endif:
Endfor: Number of Rock/Soil Types

Endcard: Aqueous Relative Permeability Card
```

A.28 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 Relative Permeability Card
SM-ML1,Burdine,,
SW1,Burdine,,
SP3,Burdine,,
SM-SP1,Burdine,,
SP2,Burdine,,
SP1,Burdine,,
US,Touma and Vauclin,1.0,2.0,
```

A.29 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 }

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 }

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 }
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),
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:

A.30 Surface Flux Card Example

```
~Surface Flux Card
3,
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,
```

A.31 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 }
    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$^d$,
        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 = { 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:
Endfor: Number Rock/Soil Types

Endcard: Aqueous Relative Permeability Card
```

A.32 Gas Relative Permeability Card 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.33 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 }
Solute Name[a],
Effective Diffusion Option[b],
{ Constant }
{ Conventional } **Note:** $D^{C}_{\ell_e} = \tau_\ell \, s_\ell \, n_D \, D^{C}_{\ell}$
{ Empirical } **Note:** Kemper and van Schaik $D^{C}_{\ell_e} = D^{C}_{\ell} \, a \, e^{(b \, n_D \, s_\ell)}$
If: Effective Diffusion Option = { Empirical }
Solute Partition Option[c],
{ Continuous [Concentration Dependent] } **Note:** $C_\ell = \dfrac{C}{n_D \, s_\ell + (1 - n_T)\rho_s \, K_{s\ell}}$
{ Noncontinous [Concentration Dependent] } **Note:** $C_\ell = \dfrac{C}{s_\ell \, n_D + s_\ell \, (1 - n_T)\rho_s \, K_{s\ell}}$
IfDef: Radioactive

```
      Half-Life$^d$, Units$^e$ (s),
      Format: Char$^a$, Char$^b$, Char$^c$, Real$^d$, Char$^e$,
    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 [ Concentration Dependent ] }  Note:  $C_\ell = \frac{C}{n_D s_\ell + (1 - n_T) \rho_s K_{s\ell}}$
    { Noncontinous [ Concentration Dependent ] }  Note:  $C_\ell = \frac{C}{s_\ell n_D + s_\ell (1 - n_T) \rho_s K_{s\ell}}$
    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 }
  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 }
    { Temperature Dependent }  Note:  $\ln(K_{g\ell}) = a + \frac{b}{T} + c \ln(T) + d\,T + e\,T^2$
  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 }
        a$^g$, b$^h$, c$^i$, d$^j$, 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:
```

Endcard:

A.34 Solute/Fluid Interactions 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.35 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: Solute Partition Option: Solute/Fluid Interactions Card = { Concentration Dependent }
        Dependent Solute Name[d],
        "a" Parameter[e], "b" Parameter[f],
        If: Effective Diffusion Option: Solute/Fluid Interactions Card = { Empirical }
            Aqueous Molecular Diffusion Coefficient[d], Units[e] (m^2/s), a[f], b[g],
            Format: Char[a], Real[b], Char[c], Char[d], Real[e], Real[f], Real[g], Char[h], Real[i], Real[j],
        Else:
            Format: Char[a], Real[b], Char[c], Char[d], Real[e], Real[f]
        Endif:
    Else:
        If: Effective Diffusion Option: Solute/Fluid Interactions Card = { Empirical }
            Aqueous Molecular Diffusion Coefficient[d], Units[e] (m^2/s), a[f], 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:
    Endif:
Endfor: Number of Solutes
```

```
Endfor: Number of Rock/Soil Types
Endcard:
```

A.36 Solute/Porous Media Interaction Card Examples

```
~Solute/Porous Media Interaction Card
SP1,,,,,
TCE,0.0,m^3/kg,
SP2,,,,,
TCE,0.0,m^3/kg,
SM-ML1,,,,,
TCE,0.0,m^3/kg,
SM-SP1,,,,,
TCE,0.0,m^3/kg,
SP3,,,,,
TCE,0.0,m^3/kg,
SW1,,,,,
TCE,0.0,m^3/kg,
US,,,,,
TCE,0.0,m^3/kg,
```

```
~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,,,,,
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.37 Reactive Transport Control Card

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

Reactive Transport Execution Mode Option[a]
{ Normal }
Format: *Char[a],*

Reactive Transport Solution Mode Option[a]
{ Fully Coupled }
{ Operator Split }
Format: *Char[a],*

Aitken extrapolation scheme switch[a], Column scan switch[b], Non-reacting solver switch[c], Effective reaction area switch[d], Texture solver / transport solver coupling switch[e], Porosity update with precipitation/dissolution[f], Lower boundary condition switch (if 0, dC/dz=0, if 1, C=0)[g], Water consumption in chemical reactions / water flow coupling switch[h], Gas consumption in chemical reactions / gas flow coupling switch[i]
Format: Integer (0 or 1)[a], Integer (0 or 1)[b], Integer (0 or 1)[c], Integer (0 or 1)[d], Integer (0 or 1)[e], Integer (0 or 1)[f], Integer (0 or 1)[g], Integer (0 or 1)[h], Integer (0 or 1)[i]

Max. CFL[a], Crank-Nicolson parameter[b], Error in non-reacting species[c], Error in equation point solver[d], Error in transport point solver[e], Error in texture solver[f], Error in normal concentration solver[g]
Format: Real[a], Real[b], Real[c], Real[d], Real[e], Real[f], Real[g]

A.38 Reactive Transport Control Card Example

```
~Reactive Transport Control card
Normal,
Fully Coupled,
0,0,0,0,1,0,0,0,0,
2.0E-06, 1.0E+00, 1.0E-06, 1.0E-07, 1.0E-03, 5.0E-03, 2.0E-03,
```

A.39 Aqueous Species Card

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

Number of Aqueous Species[a]
Format: Integer[a]

For: Number of Aqueous Species

> Name of Aqueous Species[a], Charge[b], Hard Core Diameter[c], Number of Elements[d]
> **Format:** Char[a], Real[b], Real[c], Integer[d]
> **For:** Number of Elements in Aqueous Species
> Stoichiometric Coefficient of Element[a], Element Name[b]
> Format: Real[a], Char[b],
> **Endfor:** Number of Elements in Aqueous Species

Endfor: Number of Aqueous Species

Endcard:

A.40 Aqueous Species Card Example

```
~Aqueous Species card
 6,
'H+          ', 1.0000, 9.0000, 1,
 1.0000,' H',
'H2O         ', 0.0000,-4.0000, 2,
 2.0000,' H',
 1.0000,' O',
'OH-         ',-1.0000, 3.0000, 2,
 1.0000,' H',
 1.0000,' O',
```

```
'CO2(aq)      ', 0.0000, 0.0000, 2,
 1.0000,' C',
 2.0000,' O',
'HCO3-        ',-1.0000, 4.0000, 3,
 1.0000,' H',
 1.0000,' C',
3.0000,' O',
'TcO4(1-)     ',-1.0000, 4.0000, 2,
 1.0000,'Tc',
 4.0000,' O',
```

A.41 Gas Species Card

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

Number of Gas Species[a]
Format: Integer[a]

For: Number of Gas Species

```
Name of Gas Species[a], Name of Equilibrium Aqueous Species[b], Gas Species Concentration
    Dependency (fixed or transient)[c], Gas Diffusion Coefficient (m^2 s^-1)[d],
    Number of Elements in Gaseous Species[e]
Format: Char[a], Char[b], Char[c], Real[d], Integer[e]
For: Number of Elements in Gas Species
    Stoichiometric Coefficient of Element[a], Element Name[b]
    Format: Real[a], Char[b],
Endfor: Number of Elements in Gas Species
Number of Terms in Henry's Coefficient vs. Temperature Curve Fit[a]
Format: Integer[a]
For: Number of Terms in Henry's Coefficient vs. Temperature Curve Fit
log Henry's Law Equilibrium Coefficient Term (M/atm)[a],
Format: Real[a]
Endfor: Number of Terms in Henry's Coefficient vs. Temperature Curve Fit
```

Endfor: Number of Gas Species

Endcard:

A.42 Gas Species Card Example

```
~Gas Species card
   2,
```

```
'CO2(g)      ', 'CO2(aq)      ', 'transient',  1.69443E-06, 2,
 1.0000,' C',
 2.0000,' O',
 4,
-1.096100E+00,-1.796793E-02, 1.321931E-04,-3.982381E-07,
'O2(g)       ', 'O2(aq)       ', 'fixed',  2.40000E-06, 1,
 2.0000,' O',
 1,
-2.656700E+00,
```

A.43 Components Card

Card Title[a] { ~Components [Card] }
Format: *Char[a]*
Number of Components[a]
Format: Integer[a]

For: Number of Components

Name of Component[a], Molecular Mass (kg kmol^{-1})[b], Number of Aqueous Species[c],
Format: Char[a], Real[b], Integer[c]
For: Number Aqueous Species
Stoichiometric Coefficient of Aqueous Species[a], Aqueous Species Name[b]
Format: Real[a], Char[b],
Endfor: Number of Aqueous Species

Endfor: Number of Aqueous Species

Endcard:

A.44 Components Card Example

```
~Components card
3,
'B(total)    ',10.8110,  2,
 1.0000,'BO2-        ',
 1.0000,'CaCO3(aq)   ',
'Tc(total)   ',98.0000,  1,
 1.0000,'TcO4(1-)    ',
'CO2(total)  ',44.0100,  3,
 1.0000,'CO2(aq)     ',
 1.0000,'HCO3-       ',
 1.0000,'CO3(2-)     ',
```

A.45 Solid Species Card

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

Number of Solid Species[a]
Format: Integer[a]

For: Number of Solid Species

> Name of Solid Species[a], Mass Density (Mg m^{-3}), Number of Elements[c]
> **Format:** Char[a], Real[b], Real[c], Integer[d]
> **For:** Number of Elements in Solid Species
>   Stoichiometric Coefficient of Element[a], Element Name[b]
>   Format: Real[a], Char[b],
> **Endfor:** Number of Elements in Solid Species

Endfor: Number of Solid Species

Endcard:

A.46 Solid Species Card Example

```
~Solid Species card
   4,
'Quartz      ', 2.6480,  2,
```

```
 1.0000,'Si',
 2.0000,' O',
'Albiteh    ', 2.6100,  4,
 1.0000,'Al',
 1.0000,'Na',
 3.0000,'Si',
 8.0000,' O',
'K-feldspar ', 2.5565,  4,
 1.0000,'Al',
 1.0000,' K',
 3.0000,'Si',
 8.0000,' O',
'illite     ', 2.7631,  6,
 0.6000,' K',
 0.2500,'Mg',
 2.3000,'Al',
 3.5000,'Si',
12.0000,' O',
 2.0000,' H',
```

A.47 Fast Reactions Card

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

Number of Fast Reactions [a]
Format: Integer[a]

For: Number of Fast Reactions

Number of Aqueous Species[a]
Format: Integer[a]
For: Number Aqueous Species
    Stoichiometric Coefficient of Aqueous Species[a], Aqueous Species Name[b]
    Format: Real[a], Char[b],
Endfor: Number of Aqueous Species

Number of Coefficients in Equilibrium Constant vs. temperature polynomial function[a], Exponential Factor of Equilibrium Reaction[b]
Format: Integer[a], Real[b]
For: Number of Coefficients in Equilibrium Constant vs. temperature polynomial function
    Coefficient in Equilibrium Constant vs. temperature polynomial function[a]
Format: Real[a], Real[a], Real[a], ...
Endfor:

Endfor: Number of Fast Reactions

Endcard:

A.48 Fast Reactions Card Example

```
~Fast Reactions card
   3,
   3,
-1.0000,'H2O          ',
 1.0000,'H+           ',
 1.0000,'OH-          ',
   5,    1.00000,
-1.49400E+01, 4.17100E-02,-1.91700E-04, 4.95000E-07,-6.15900E-10,
   4,
-1.0000,'H2O          ',
-1.0000,'CO2(aq)      ',
 1.0000,'H+           ',
 1.0000,'HCO3-        ',
   5,    1.00000,
-6.57600E+00, 1.19800E-02,-1.36200E-04, 4.13100E-07,-5.48300E-10,
   3,
-1.0000,'HCO3-        ',
 1.0000,'H+           ',

 1.0000,'CO3(2-)      ',
   5,    1.00000,
-1.06200E+01, 1.39700E-02,-1.14300E-04, 3.35200E-07,-4.68700E-10,
```

A.49 Kinetic Aqueous Reactions Card

Card Title[a] { ~Kinetic Aqueous Reactions [Card] }
Format: *Char*[a]
Number of Kinetic Aqueous Reactions[a]
Format: Integer[a]

For: Number of Kinetic Aqueous Reactions
Endfor:

Endcard:

A.50 Kinetic Aqueous Reactions Card Example

```
~Kinetic Aqueous Reactions card
   0,
```

A.51 Slow Reactions Card

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

Number of Slow Reactions[a]
Format: Integer[a]

For: Number of Slow Reactions

Mass Action Law Type[a] {Full = 1} {Reduced = -1} {Glass = 0},
Number of Solid Species in Slow Reaction[b] {1 or 2}
Format: Integer[a], Integer[b]
For: Number of Solid Species in Slow Reaction
Stoichiometric Coefficient of Solid Species in Slow Reaction[a], Name of Solid Species[b]
Format: Real[a], Character[b]
EndFor: Number of Solid Species in Slow Reaction
Number Aqueous Species[a]
Format: Integer[a]
For: Number Aqueous Species
Stoichiometric Coefficient of Aqueous Species[a], Aqueous Species Name[b],
Species Participation in Reduced Mass Action Reaction Rate Calculation[c],
{ -1 = Forward Rate Only}, { 0 = Neither }, { 1 = Equilibrium Only }
Stoichiometric Coefficient of Aqueous Species Used in Equilibrium Calculations[d]
Format: Real[a], Char[b], Real[c], Real[d],
Endfor: Number of Aqueous Species
Rate Constant of Solid Reaction[a], Activation Energy of Solid Reaction[b], Exponential Factor of Solid Reaction[c], Exponential Factor of pH dependence of dissolution Reaction[d], Number of Coefficients in Equilibrium Constant vs. temperature polynomial Function[e],
Format: Real[a], Real[b], Real[c], Real[d], Integer[e]
For: Number of Coefficients in Equilibrium Constant vs. temperature polynomial function
Coefficient in Equilibrium Constant vs. temperature polynomial function[a]
Format: Real[a], Real[a], Real[a], ...
Endfor:

Endfor: Number of Slow Reactions

Endcard:

A.52 Slow Reactions Card Example

```
~Slow Reactions card
   2,
 0,1,
-1.0000,'glass       ',
   8,
-4.50000E-01,'H+          ', 0.0000, 0.0000,
 2.02000E-02,'K+          ', 0.0000, 0.0000,
 4.20800E-01,'Na+         ', 0.0000, 0.0000,
 4.60000E-02,'Ca2+        ', 0.0000, 0.0000,
 1.54000E-01,'Al(OH)3(aq) ', 0.0000, 0.0000,
 6.06000E-01,'SiO2(aq)    ', 1.0000, 1.0000,
 9.40000E-02,'BO2-        ', 0.0000, 0.0000,
 1.61720E-06,'TcO4(1-)    ', 0.0000, 0.0000,
   0.5594027,  74.8,   1.00000, -0.4, 1,
-3.72810E+00,
 1,1,
-1.0000,'Quartz      ',
   2,
-1.00000E+00,'OH-         ', 1.0000,-1.0000,
 1.00000E+00,'HSiO3(1-)   ', 1.0000, 1.0000,
    -17.0052,  0.,    1.00000,  0.0, 1,
 4.82000E-02,
```

A.53 Fluid Decay Card

Card Title^a { ~Fluid Decay [Card] }
Format: *Char*^{a}

Number of Fluid Decay Reactions^a
Format: Integer^a

For: Number of Fluid Decay Reactions
Endfor:

Endcard:

A.54 Fluid Decay Card Example

```
~Fluid Decay card
   0,
```

A.55 Solid Decay Card

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

Number of Solid Decay Reactions[a]
Format: Integer[a]

For: Number of Solid Decay Reactions
Endfor:

Endcard:

A.56 Solid Decay Card Example

```
~Solid Decay card
   0,
```

A.57 Lithology Card

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

For: Number of Rock/Soil Types

- Rock/Soil Name[a], Number of Solid Species[b],
- **Format:** Char[a], Integer[b],
- **For:** Number of Solid Species
  - Name of Solid Species[a], Relative Solid Volume[b], Radius[c]
  - Format: Char[a], Real[b], Real[c]
- **Endfor:**

Endfor:

Endcard:

A.58 Lithology Card Example

```
~Lithology card
'Backfill Soil',  9,
'glass        ', 0.00000E+00, 0.50000E-02,
'Quartz       ', 0.39474E+00, 0.10000E-03,
'Albiteh      ', 0.39474E+00, 0.10000E-03,
'K-feldspar   ', 0.98684E-01, 0.10000E-03,
'illite       ', 0.98684E-01, 0.50000E-05,
'analcime     ', 0.32895E-02,-0.20000E-03,
'calcite      ', 0.32895E-02,-0.20000E-03,
'chalcedony   ', 0.32895E-02,-0.20000E-03,
'NaAlSi2O6.6H', 0.32895E-02,-0.20000E-03,
'Glass Waste ',  9,
'glass        ', 0.98870E+00, 0.50000E-02,
'Quartz       ', 0.00000E+00, 0.10000E-03,
'Albiteh      ', 0.00000E+00, 0.10000E-03,
'K-feldspar   ', 0.00000E+00, 0.10000E-03,
'illite       ', 0.00000E+00, 0.50000E-05,
'analcime     ', 0.28249E-02,-0.20000E-03,
'calcite      ', 0.28249E-02,-0.20000E-03,
'chalcedony   ', 0.28249E-02,-0.20000E-03,
'NaAlSi2O6.6H', 0.28249E-02,-0.20000E-03,
```

A.59 Reaction BC Card

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

Number of Boundary Segments [a]
Format: Integer[a]

For: Number of Boundary Segments
Boundary Segment Number[a], Boundary Condition Type[b] { 1 = Diriclet }
Format: Integer[a], Integer[b]

Number of Solids in Equilibrium with Solution[a],
Format: Integer[a],
For: Number of Solids in Equilibrium with Solution,

Reaction Number of Solid Reaction[a]
Format: Integer[a],
Endfor:

Number of Kinetic Reactions in Equilibrium with Solution[a],
Format: Integer[a],
For: Number of Kinetic Reactions in Equilibrium with Solution,
Reaction Number of Kinetic Reaction[a]
Format: Integer[a],
Endfor:

Number of Specified Aqueous Species Concentrations[a]
Format: Integer[a],
For: Number of Specified Aqueous Species Concentrations,
Specifiec Total Concentration[a], Number of Aqueous Species[b]
Format: Real[a], Integer[b]
Stoichiometric Coefficient of Aquous Species[a], Name of Aqueous Species[b]
Format: Real[a], Char[a],
Endfor:
Endfor:

Endcard:

A.60 Reaction BC Card Example

```
~Reaction BC card
   1,
1,1,
   0,
   0,
   12,
  0.10000E+01,  1,
 1.0000,'H2O          ',
  0.10000E-06,  1,
 1.0000,'H+           ',
  0.15900E-03,  1,
 1.0000,'CO2(aq)      ',
  0.10000E-05,  2,
 1.0000,'Al(OH)3(aq) ',
 1.0000,'Al(OH)4-     ',
  0.10000E-04,  1,
 1.0000,'SiO2(aq)     ',
  0.10000E-05,  1,
```

```
1.0000,'K+           ',
 0.10000E-04,  1,
1.0000,'Na+          ',
 0.10000E-06,  1,
1.0000,'Ca2+         ',
 0.10000E-08,  1,
1.0000,'BO2-         ',
 0.32000E-03,  1,
1.0000,'O2(aq)       ',
 0.10000E-08,  1,
1.0000,'Mg(2+)       ',
 0.10000E-11,  1,
1.0000,'TcO4(1-)     ',
```

A.61 Reaction IC card

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

Number of Reaction Initial Conditions[a]
Format: Integer[a]

For: Number of Reaction Initial Conditions
Endfor:

Endcard:

A.62 Reaction IC Card Example

```
~Reaction IC card
   0,
```

Note: the default is for the initial conditions to be equal to the inlet boundary conditions.

Distribution

No. of
Copies

OFFSITE

Argonne National Laboratory
Building 205
9700 South Cass Avenue
Argonne, IL 60439-4837
ATTN: W. L. Ebert

Lawrence Berkeley National Laboratory
Earth Sciences Division
University of California
Berkeley, CA 94720
ATTN: K. Pruess

2 Lawrence Livermore National Laboratory
University of California
7000 East Avenue
Livermore, CA 94550-9234
ATTN: T. J. Wolery, Mailcode L-219
W. E. Glassley, Mailcode L-202

Penn State University
College of Engineering
0212 Sackett Building
University Park, PA 16802
ATTN: G.-T. Yeh

4 Yucca Mountain Site Characterization Program
TRW Environmental Safety System
1180 Town Center Drive
Mail Stop 423
Las Vegas, NV 89134
ATTN: R. Andrews
E. Bartlett Mann
J. Lee
Y. Chen

No. of
Copies

ONSITE

6 DOE Richland Operations Office

C. A. Babel	H6-60
N. R. Brown	H6-60
P. E. Lamont	H6-60
Public Reading Room	H2-53
RL Docket File (2)	H5-36

3 Bechtel Hanford, Inc.

G. A. Jewell (2)	H0-21
B. H. Ford	H0-19

2 Fluor Hanford

A. M. Umek	H8-49
T. B. Veneziano	H8-66

Lockheed Martin Hanford Corporation

J. A. Voogd	H6-64

11 Fluor Federal Services

D. A. Burbank	S4-45
K. C. Burgard	S4-45
T. A. Carlson	B4-09
E. J. Freeman	B4-43
R. Khaleel	B4-43
F. M. Mann (2)	H0-22
M. G. Piepho	R3-26
R. J. Puigh	B4-43
R. W. Root	R2-53
P. D. Rittmann	B4-43

No. of Copies		
38	Pacific Northwest National Laboratory	
	M. P. Bergeron	K9-36
	R. W. Bryce	K6-75
	D. H. Bacon (10)	K9-33
	A. Chilakapati	K9-36
	M. J. Fayer	K9-33
	N. D. Foote	K8-14
	C. T. Kincaid	K9-33
	B. P. McGrail (10)	K6-81
	W. E. Nichols	K9-33
	K. P. Saripalli	K6-81
	J. L. Smoot	K8-46
	M. D. White	K9-36
	S. B. Yabusaki	K9-36
	Information Release Office (7)	K1-06