LA-UR-06-7048

Approved for public release;
distribution is unlimited.

Title:	*Quick Reference Guide: PFLOTRAN 2.0 (LA-CC-09-047)* *Multiphase-Multicomponent-Multiscale Massively Parallel* *Reactive Transport Code*
Author(s):	SciDAC-2 Project (PI: Peter C. Lichtner, lichtner@lanl.gov) Glenn Hammond (glenn.hammond@pnnl.gov)
Contacts:	Richard Mills (rmills@ornl.gov) Chuan Lu (Chuan.Lu@inl.gov) Bobby Philip (bphilip@lanl.gov) Barry Smith (bsmith@anl.gov)
Date:	March 29, 2011

DRAFT

Los Alamos
NATIONAL LABORATORY

TABLE OF CONTENTS

1 Introduction

PFLOTRAN solves a system of generally nonlinear partial differential equations describing multiphase, multicomponent and multiscale reactive flow and transport in porous materials. The code is designed to run on massively parallel computing architectures as well as workstations and labtops (Hammond et al., 2011). Parallelization is achieved through domain decomposition using the PETSc (Portable Extensible Toolkit for Scientific Computation) libraries for the parallelization framework (Balay et al., 1997).

2 Governing Equations

2.1 Mode: `RICHARDS`

`RICHARDS` Mode applies to single phase, variably saturated, isothermal systems. The governing mass conservation equation is given by

$$\frac{\partial}{\partial t}\left(\varphi s \rho\right) + \nabla\cdot\left(\rho \boldsymbol{q}\right) = Q_w, \tag{1}$$

and

$$\boldsymbol{q} = -\frac{k k_r}{\mu}\nabla\left(P - \rho g z\right). \tag{2}$$

Here, φ denotes porosity, s saturation, ρ water density, $\boldsymbol{q}$ Darcy flux, k intrinsic permeability, k_r relative permeability, μ viscosity, P pressure, g gravity, and z the vertical component of the position vector. Supported relative permeability functions k_r for Richards' equation include van Genuchten, Books-Corey and Thomeer-Corey, while the saturation functions include Burdine and Mualem. Water density and viscosity are computed as a function of temperature and pressure through an equation of state for water.

2.2 Mode: `MPHASE`

Local equilibrium is assumed between phases for modeling multiphase systems with PFLOTRAN. The multiphase partial differential equations for mass and energy conservation solved by PFLOTRAN have the general form:

$$\frac{\partial}{\partial t}\left(\varphi\sum_\alpha s_\alpha\rho_\alpha X_i^\alpha\right) + \nabla\cdot\sum_\alpha\left[\boldsymbol{q}_\alpha\rho_\alpha X_i^\alpha - \varphi s_\alpha D_\alpha\rho_\alpha\nabla X_i^\alpha\right] \;=\; Q_i, \tag{3a}$$

for the ith component, and

$$\frac{\partial}{\partial t}\left(\varphi\sum_\alpha s_\alpha\rho_\alpha U_\alpha + (1-\varphi)\rho_r c_r T\right) + \nabla\cdot\sum_\alpha\left[\boldsymbol{q}_\alpha\rho_\alpha H_\alpha - \kappa\nabla T\right] \;=\; Q_e. \tag{3b}$$

for energy. In these equations α designates a fluid phase (e.g. H_2O, supercritical CO_2) at temperature T and pressure P_α with the sums over all fluid phases present in the system; species are designated by the subscript i (e.g. $w = \mathrm{H_2O}$, $c = \mathrm{CO_2}$); φ denotes the porosity of the geologic formation; s_α denotes the phase saturation state; X_i^α denotes the mole fraction of species i ($\sum_i X_i^\alpha = 1$); ρ_α, H_α, U_α refer to the molar density, enthalpy, and internal energy of each fluid phase, respectively; and $\boldsymbol{q}_\alpha$ denotes the Darcy flow rate defined by

$$\boldsymbol{q}_\alpha = -\frac{kk_\alpha}{\mu_\alpha}\nabla\big(P_\alpha - W_\alpha\rho_\alpha g\boldsymbol{z}\big), \tag{4}$$

where k refers to the intrinsic permeability, k_α denotes the relative permeability, μ_α denotes the fluid viscosity, W_α denotes the formula weight, g denotes the acceleration of gravity, and z designates the vertical of the position vector. The source/sink terms, Q_i and Q_e, describe injection and extraction of mass and heat at wells, respectively. The quantities ρ_r, c_r, and κ refer to the density, heat capacity, and thermal conductivity of the porous rock.

Additional constitutive relations are needed to account for capillary pressure, and changes in phase which are not discussed in detail here (see [**?**]). In PFLOTRAN a variable switching approach is used to account for phase changes enforcing local equilibrium. According to the Gibbs phase rule there are a total of N_C+1 degrees of freedom where N_C denotes the number of independent components. This can be seen by noting that the intensive degrees of freedom are equal to $N_{\rm int} = N_C - N_P + 2$, where N_P denotes the number of phases. The extensive degrees of freedom equals $N_{\rm ext} = N_P - 1$. This gives a total number of degrees of freedom $N_{\rm dof} = N_{\rm int} + N_{\rm ext} = N_C + 1$, independent of the number of phases N_P in the system.

2.3 Mode: `IMMIS`

The `IMMIS` mode applies to multiple immiscible phases.

2.4 Mode: `Air-Water`

The `Air-Water` mode involves two phase liquid water-gas flow coupled to the reactive transport mode. Mass conservation equations have the form

$$\frac{\partial}{\partial t}\varphi\Big(s_l\rho_l x_i^l + s_g\rho_g x_i^g\Big) + \nabla\cdot\Big(\boldsymbol{q}_l\rho_l x_i^l + \boldsymbol{q}_g\rho_g x_i^g - \varphi s_l D_l\rho_l\nabla x_i^l - \varphi s_g D_g\rho_g\nabla x_i^g\Big) = Q_i, \tag{5}$$

for liquid and gas saturation $s_{l,g}$, density $\rho_{l,g}$, diffusivity $D_{l,g}$, Darcy velocity $\boldsymbol{q}_{l,g}$ and mole fraction $x_i^{l,g}$. The energy conservation equation can be written in the form

$$\sum_{\alpha=l,g}\left\{\frac{\partial}{\partial t}\big(\varphi s_\alpha\rho_\alpha U_\alpha\big) + \nabla\cdot\big(\boldsymbol{q}_\alpha\rho_\alpha H_\alpha\big)\right\} + \frac{\partial}{\partial t}\Big((1-\varphi)\rho_r C_p T\Big) - \nabla\cdot(\kappa\nabla T)\Big) = Q, \tag{6}$$

as the sum of contributions from liquid and gas fluid phases and rock, with internal energy U_α and enthalpy H_α of fluid phase α, rock heat capacity C_p and thermal conductivity κ. Note that

$$U_\alpha = H_\alpha - \frac{P_\alpha}{\rho_\alpha}. \tag{7}$$

2.5 Mode: THC (Thermal-Hydrologic-Chemical)

THC Mode applies to single phase, variably saturated, nonisothermal systems with incorporation of density variations coupled to fluid flow. The THC equations may be coupled to the reactive transport mode (see section **??**). The governing equations for mass and energy are given by

$$\frac{\partial}{\partial t}\big(\varphi s \rho\big) + \nabla\cdot\big(\rho \boldsymbol{q}\big) = Q_w, \tag{8}$$

and

$$\frac{\partial}{\partial t}\big(\varphi s \rho U + (1-\varphi) c_p T\big) + \nabla\cdot\big(\rho \boldsymbol{q} H - \kappa \nabla T\big) = Q_e, \tag{9}$$

with the Darcy flow velocity $\boldsymbol{q}$ given by

$$\boldsymbol{q} = -\frac{k k_r}{\mu}\nabla\big(P - \rho g z\big). \tag{10}$$

Here, φ denotes porosity, s saturation, ρ mixture density of the brine, $\boldsymbol{q}$ Darcy flux, k intrinsic permeability, k_r relative permeability, μ viscosity, P pressure, g gravity, and z the vertical component of the position vector. Supported relative permeability functions k_r for Richards' equation include van Genuchten, Books-Corey and Thomeer-Corey, while the saturation functions include Burdine and Mualem. Water density and viscosity are computed as a function of temperature and pressure through an equation of state for water. The quantity κ denotes the thermal conductivity of the porous medium-fluid system. The internal energy and enthalpy of the fluid, U and H, are obtained from an equation of state for pure water. These two quantities are related by the thermodynamic expression

$$U = H - \frac{P}{\rho}. \tag{11}$$

Nonreactive solute transport equations representing e.g. NaCl have the form

$$\frac{\partial}{\partial t}\varphi s \rho x_i + \nabla\cdot\Big(\boldsymbol{q}\rho x_i - \varphi D \rho \nabla x_i\Big) = Q_i, \tag{12}$$

with mole fraction x_i, source term Q_i, and diffusion/dispersion coefficient D. Summing this equation over all components i using $\sum_i x_i = 1$, leads to Eqn.(**??**) with

$$Q_w = \sum_i Q_i. \tag{13}$$

Additional constitutive relations are needed to close the set of governing equations.

2.6 Mode: Reactive Transport (Keyword CHEMISTRY)

The governing mass conservation equations for the geochemical transport mode for a multiphase system written in terms of a set of independent aqueous primary or basis species with the form

$$\begin{aligned}\frac{\partial}{\partial t}\Big(\varphi \sum_\alpha s_\alpha \Psi_j^\alpha\Big) + \nabla\cdot\sum_\alpha\big(\boldsymbol{q}_\alpha - \varphi s_\alpha \boldsymbol{D}_\alpha \nabla\big)\Psi_j^\alpha \\ = Q_j - \sum_m \nu_{jm} I_m - \frac{\partial S_j}{\partial t},\end{aligned} \tag{14}$$

where the sums over α are over all fluid phases in the system, and where Ψ_j^α denotes the total concentration of the jth primary species $\mathscr{A}_j^{\rm pri}$ in the αth fluid phase defined by

$$\Psi_j^\alpha = \delta_{l\alpha} C_j^l + \sum_{i=1}^{N_{\rm sec}} \nu_{ji}^\alpha C_i^\alpha. \tag{15}$$

In this equation the subscript l represents the aqueous electrolyte phase from which the primary species are chosen. The secondary species concentrations C_i^α are obtained from mass action equations corresponding to equilibrium conditions of the reactions

$$\sum_j \nu_{ji}^\alpha \mathscr{A}_j^l \rightleftharpoons \mathscr{A}_i^\alpha, \tag{16}$$

yielding

$$C_i^\alpha = \frac{K_i^\alpha}{\gamma_i^\alpha} \prod_j \left(\gamma_j^l C_j^l\right)^{\nu_{ji}^\alpha}, \tag{17}$$

with equilibrium constant K_i^α, and activity coefficients γ_k^α.

The quantity Q_j denotes a source/sink term and S_j represents the sorbed concentration of the jth primary species. Diffusion is assumed to be species independent, but may be different for different phases, e.g. an aqueous electrolyte solution or gas phase. Dispersivity currently must be described through a diagonal dispersion tensor. The Darcy velocity $\boldsymbol{q}_\alpha$ for phase α is given by

$$\boldsymbol{q}_\alpha = -\frac{k k_\alpha}{\mu_\alpha} \nabla\left(p_\alpha - \rho_\alpha g z\right), \tag{18}$$

with bulk permeability of the porous medium k and relative permeability k_α, fluid viscosity μ_α, pressure p_α, density ρ_α, and acceleration of gravity g.

2.6.1 Sorption

Sorption reactions incorporated into PFLOTRAN consist of ion exchange and surface complexation reactions for both equilibrium and multirate formulations.

2.6.1.1 Ion Exchange Ion exchange reactions may be represented either in terms of bulk- or mineral-specific rock properties. Changes in bulk sorption properties can be expected as a result of mineral reactions. However, only the mineral-based formulation enables these effects to be captured in the model. The bulk rock sorption site concentration ω_α, in units of moles of sites per bulk sediment volume (mol/dm^3), is related to the bulk cation exchange capacity Q_α (mol/kg) by the expression

$$\omega_\alpha = \frac{N_{\rm site}}{V} = \frac{N_{\rm site}}{M_s}\frac{M_s}{V_s}\frac{V_s}{V} = Q_\alpha \rho_s (1-\phi). \tag{19}$$

The cation exchange capacity associated with the mth mineral is defined on a molar basis as

$$\omega_m^{\rm CEC} = \frac{N_m}{V} = \frac{N_m}{M_m}\frac{M_m}{V_m}\frac{V_m}{V} = Q_m^{\rm CEC} \rho_m \phi_m. \tag{20}$$

Ion exchange reactions are expressed in the form

$$z_i\mathscr{A}_j + z_j X^\alpha_{z_i}\mathscr{A}_i \rightleftharpoons z_j\mathscr{A}_i + z_i X^\alpha_{z_j}\mathscr{A}_j, \tag{21}$$

with valencies z_j, z_i of cations $\mathscr{A}_j$ and $\mathscr{A}_i$, respectively. The reference cation is denoted by the subscript j and the subscript $i \neq j$ represents all other cations. The corresponding mass action equation is given by

$$K^\alpha_{ji} = \frac{(k^\alpha_j)^{z_i}}{(k^\alpha_i)^{z_j}} = \left(\frac{X^\alpha_j}{a_j}\right)^{z_i}\left(\frac{a_i}{X^\alpha_i}\right)^{z_j}. \tag{22}$$

Using the Gaines-Thomas convention, the equivalent fractions X^α_k are defined by

$$X^\alpha_k = \frac{z_k S^\alpha_k}{\sum\limits_l z_l S^\alpha_l} = \frac{z_k}{\omega_\alpha} S^\alpha_k, \tag{23}$$

with

$$\sum_k X^\alpha_k = 1. \tag{24}$$

The site concentration ω_α is defined by

$$\omega_\alpha = \sum_k z_k S^\alpha_k, \tag{25}$$

where ω_α is related to the cation exchange capacity Q_α (CEC) by the expression

$$\omega_\alpha = (1-\varphi)\rho_s Q_\alpha, \tag{26}$$

with solid density ρ_s and porosity φ.

For equivalent exchange ($z_j = z_i = z$), an explicit expression exists for the sorbed concentrations given by

$$S^\alpha_j = \frac{\omega_\alpha}{z}\frac{k^\alpha_j \gamma_j m_j}{\sum\limits_l k^\alpha_l \gamma_l m_l}, \tag{27}$$

where m_k denotes the kth cation molality. This expression follows directly from the mass action equations and conservation of exchange sites.

In the more general case ($z_i \neq z_j$) it is necessary to solve the nonlinear equation

$$X^\alpha_j + \sum_{i\neq j} X^\alpha_i = 1, \tag{28}$$

for the reference cation mole fraction X_j. From the mass action equation Eqn.(**??**) it follows that

$$X^\alpha_i = k^\alpha_i a_i \left(\frac{X^\alpha_j}{k^\alpha_j a_j}\right)^{z_i/z_j}. \tag{29}$$

Defining the function

$$f(X^\alpha_j) = X^\alpha_j + \sum_{i\neq j} X^\alpha_i(X^\alpha_j) - 1, \tag{30}$$

its derivative is given by

$$\frac{df}{dX_j^\alpha} = 1 - \frac{1}{z_j X_j^\alpha} \sum_{i \neq j} z_i k_i^\alpha a_i \left(\frac{X_j^\alpha}{k_j^\alpha a_j}\right)^{z_i/z_j}. \tag{31}$$

The reference mole fraction is then obtained by Newton-Raphson iteration

$$(X_j^\alpha)^{k+1} = (X_j^\alpha)^k - \frac{f[(X_j^\alpha)^k]}{\dfrac{df[(X_j^\alpha)^k]}{dX_j^\alpha}}. \tag{32}$$

The sorbed concentration for the jth cation appearing in the accumulation term is given by

$$S_j^\alpha = \frac{\omega_\alpha}{z_j} X_j^\alpha, \tag{33}$$

with the derivatives for $j \neq l$

$$\frac{\partial S_j^\alpha}{\partial m_l} = -\frac{\omega_\alpha}{m_l} \frac{X_j^\alpha X_l^\alpha}{\sum\limits_l z_l X_l^\alpha}, \tag{34a}$$

$$= -\frac{1}{m_l} \frac{z_j z_l S_j^\alpha S_l^\alpha}{\sum\limits_l z_l^2 S_l^\alpha}, \tag{34b}$$

and for $j = l$

$$\frac{\partial S_j^\alpha}{\partial m_j} = \frac{\omega_\alpha X_j^\alpha}{z_j m_j} \left(1 - \frac{z_j X_j^\alpha}{\sum\limits_l z_l X_l^\alpha}\right), \tag{35a}$$

$$= \frac{S_j^\alpha}{m_j} \left(1 - \frac{z_j^2 S_j^\alpha}{\sum\limits_l z_l^2 S_l^\alpha}\right). \tag{35b}$$

2.6.1.2 Surface Complexation Surface complexation reactions are assumed to have the form

$$\nu_\alpha >\chi_\alpha + \sum_j \nu_{ji} \mathscr{A}_j \rightleftharpoons >\mathscr{S}_{i\alpha}, \tag{36}$$

for the ith surface complex $>\mathscr{S}_{i\alpha}$ on site α and empty site $>\chi_\alpha$. As follows from the corresponding mass action equation the equilibrium sorption concentration $S_{i\alpha}^{\rm eq}$ is given by

$$S_{i\alpha}^{\rm eq} = \frac{\omega_\alpha K_i Q_i}{1 + \sum_l K_l Q_l}, \tag{37}$$

and the empty site concentration by

$$S_\alpha^{\rm eq} = \frac{\omega_\alpha}{1 + \sum_l K_l Q_l}, \tag{38}$$

where the ion activity product Q_i is defined by

$$Q_i = \prod_j \big(\gamma_j C_j\big)^{\nu_{ji}}. \tag{39}$$

The site concentration ω_α satisfies the relation

$$\omega_\alpha = S_\alpha + \sum_i S_{i\alpha}. \tag{40}$$

and is constant. The equilibrium sorbed concentration $S_{j\alpha}^{\rm eq}$ is defined as

$$S_{j\alpha}^{\rm eq} = \sum_i \nu_{ji} S_{i\alpha}^{\rm eq} = \frac{\omega_\alpha}{1+\sum_l K_l Q_l} \sum_i \nu_{ji} K_i Q_i. \tag{41}$$

2.6.1.3 Multirate Sorption In the multirate model the rates of sorption reactions are described through a kinetic relation given by

$$\frac{\partial S_{i\alpha}}{\partial t} = k_\alpha\big(S_{i\alpha}^{\rm eq} - S_{i\alpha}\big), \tag{42}$$

for surface complexes, and

$$\frac{\partial S_\alpha}{\partial t} = -\sum_i k_\alpha\big(S_{i\alpha}^{\rm eq} - S_{i\alpha}\big), \tag{43}$$

$$= k_\alpha\big(S_\alpha^{\rm eq} - S_\alpha\big), \tag{44}$$

for empty sites, where $S_\alpha^{\rm eq}$ denotes the equilibrium sorbed concentration. For simplicity, in what follows it is assumed that $\nu_\alpha = 1$. With each site α is associated a rate constant k_α and site concentration ω_α. These quantities are defined through a given distribution of sites $\wp(\alpha)$, such that

$$\int_0^\infty \wp(k_\alpha) dk_\alpha = 1. \tag{45}$$

The fraction of sites f_α belonging to site α is determined from the relation

$$f_\alpha = \int_{k_\alpha-\Delta k_\alpha/2}^{k_\alpha+\Delta k_\alpha/2} \wp(k_\alpha) dk_\alpha \simeq \wp(k_\alpha)\Delta k_\alpha, \tag{46}$$

with the property that

$$\sum_\alpha f_\alpha = 1. \tag{47}$$

Given that the total site concentration is ω, then the site concentration ω_α associated with site α is equal to

$$\omega_\alpha = f_\alpha \omega. \tag{48}$$

An alternative form of these equations is obtained by introducing the total sorbed concentration for the jth primary species for each site defined as

$$S_{j\alpha} = \sum_i \nu_{ji} S_{i\alpha}. \tag{49}$$

Then the transport equations become

$$\frac{\partial}{\partial t}\Big(\varphi\Psi_j + \sum_\alpha S_{j\alpha}\Big) + \nabla\cdot\boldsymbol{\Omega}_j \;=\; -\sum_m \nu_{jm} I_m. \tag{50}$$

The total sorbed concentrations are obtained from the equations

$$\frac{\partial S_{j\alpha}}{\partial t} \;=\; k_\alpha\big(S_{j\alpha}^{\rm eq} - S_{j\alpha}\big). \tag{51}$$

2.6.2 Colloid-Facilitated Transport

Colloid-facilitated transport is implemented into PFLOTRAN based on surface complexation reactions. Competition between mobile and immobile colloids and stationary mineral surfaces is taken into account. Colloid filtration processes are not currently implemented into PFLOTRAN. A colloid is treated as a solid particle suspended in solution or attached to a mineral surface. Colloids may be generated through nucleation of minerals in solution, although this effect is not included currently in the code.

Three separate reactions may take place involving competition between mobile and immobile colloids and mineral surfaces

$$>X_k^{\rm m} + \sum_j \nu_{jk}\mathscr{A}_j \;\rightleftharpoons\; >S_k^{\rm m}, \tag{52}$$

$$>X_k^{\rm im} + \sum_j \nu_{jk}\mathscr{A}_j \;\rightleftharpoons\; >S_k^{\rm im}, \tag{53}$$

$$>X_k^{s} + \sum_j \nu_{jk}\mathscr{A}_j \;\rightleftharpoons\; >S_k^{s}, \tag{54}$$

with corresponding reaction rates $I_k^{\rm m}$, $I_k^{\rm im}$, and I_k^s, where the superscripts s, m, and im denote mineral surfaces, and mobile and immobile colloids, respectively. In addition, reaction with minerals $\mathscr{M}_s$ may occur according to the reaction

$$\sum_j \nu_{js}\mathscr{A}_j \;\rightleftharpoons\; \mathscr{M}_s. \tag{55}$$

The transport equations for primary species, mobile and immobile colloids, read

$$\frac{\partial}{\partial t}\varphi s_l \Psi_j^l + \nabla\cdot\boldsymbol{\Omega}_j^l \;=\; -\sum_k \nu_{jk}\big(I_k^{\rm m} + I_k^{\rm im} + I_k^s\big) - \sum_s \nu_{js} I_s, \tag{56}$$

$$\frac{\partial}{\partial t} S_k^{\rm m} + \nabla\cdot\boldsymbol{q}_c S_k^{\rm m} \;=\; I_k^{\rm m}, \tag{57}$$

$$\frac{\partial}{\partial t} S_k^{\rm im} \;=\; I_k^{\rm im}, \tag{58}$$

$$\frac{\partial}{\partial t} S_k^{s} \;=\; I_k^{s}, \tag{59}$$

where $\boldsymbol{q}_c$ denotes the colloid Darcy velocity which may be greater than the fluid velocity $\boldsymbol{q}$. For conditions of local equilibrium the sorption reaction rates may be eliminated and replaced by algebraic sorption isotherms to yield

$$\frac{\partial}{\partial t}\varphi s_l\Big[\Psi_j^l + \sum_k \nu_{jk}\big(S_k^{\rm m} + S_k^{\rm im} + S_k^s\big)\Big] + \nabla\cdot\Big(\boldsymbol{\Omega}_j^l + \boldsymbol{q}_c\sum_k \nu_{jk} S_k^{\rm m}\Big) \;=\; -\sum_s \nu_{js} I_s. \tag{60}$$

2.7 Solute Age

PFLOTRAN implements the Eulerian formulation of solute age for a nonreactive tracer following Goode (1996). PFLOTRAN solves the advection-diffusion/dispersion equation for age given by

$$\frac{\partial}{\partial t}\varphi s A C + \nabla\cdot\Big(\boldsymbol{q} AC - \varphi s D \nabla(AC)\Big) \;=\; \varphi s C, \tag{61}$$

where A denotes the mean age of the solute with concentration C. Other quantities appearing in the age equation are identical to the solute transport equation for a partially saturated porous medium with saturation state s. The age and solute transport equations are solved simultaneously.

3 Method of Solution

The flow and heat equations (Modes: RICHARDS, MPHASE, FLASH2, THC, and 2PH) are solved using a fully implicit backward Euler approach based on Newton-Krylov iteration. Both fully implicit backward Euler and operator splitting solution methods are supported for reactive transport.

3.1 Fully Implicit

In a fully implicit formulation the nonlinear equations for the residual function $\boldsymbol{R}$ given by

$$\boldsymbol{R}(\boldsymbol{x}) \;=\; \boldsymbol{0}, \tag{62}$$

are solved using an iterative solver based on the Newton-Raphson equations

$$\boldsymbol{J}^{(i)}\delta\boldsymbol{x}^{(i+1)} \;=\; -\boldsymbol{R}^{(i)}, \tag{63}$$

at the ith iteration. Iteration stops when

$$\Big|\boldsymbol{R}^{(i+1)}\Big| < \epsilon, \tag{64}$$

or if

$$\Big|\delta\boldsymbol{x}^{(i+1)}\Big| < \delta. \tag{65}$$

However, the latter criteria does not necessarily guarantee that the residual equations are satisfied. The solution is updated from the relation

$$\boldsymbol{x}^{(i+1)} \;=\; \boldsymbol{x}^{(i)} + \delta\boldsymbol{x}^{(i+1)}. \tag{66}$$

For the logarithm of the concentration with $\boldsymbol{x} = \ln \boldsymbol{y}$, the solution is updated according to the equation

$$\boldsymbol{y}^{(i+1)} \;=\; \boldsymbol{y}^{(i)} \mathrm{e}^{\delta \ln \boldsymbol{y}^{(i+1)}}. \tag{67}$$

3.2 Operator Splitting

Operator splitting involves splitting the reactive transport equations into a nonreactive part and a part incorporating reactions. This is accomplished by writing Eqns.(**??**) as the two coupled equations

$$\frac{\partial}{\partial t}\big(\varphi\sum_\alpha s_\alpha \Psi_j^\alpha\big) + \nabla\cdot\sum_\alpha \big(\boldsymbol{q}_\alpha - \varphi s_\alpha \boldsymbol{D}_\alpha \nabla\big)\Psi_j^\alpha \;=\; Q_j, \tag{68}$$

and

$$\frac{d}{dt}\big(\varphi\sum_\alpha s_\alpha \Psi_j^\alpha\big) \;=\; -\sum_m \nu_{jm} I_m - \frac{\partial S_j}{\partial t}, \tag{69}$$

The first set of equations are linear in Ψ_j (for species-independent diffusion coeffients) and solved over over a time step Δt resulting in Ψ_j^*. The result for Ψ_j^* is inverted to give the concentrations C_j^* by solving the equations

$$\Psi_j^* \;=\; C_j^* + \sum_i \nu_{ji} C_i^*, \tag{70}$$

where the secondary species concentrations C_i^* are nonlinear functions of the primary species concentrations C_j^*. With this result the second set of equations are solved implicitly for C_j at $t+\Delta t$ using Ψ_j^* for the starting value at time t.

As a simple example of operator splitting consider a single component system with retardation described by a constant K_d. According to this model the sorbed concentration S is related to the aqueous concentration by the linear equation

$$S \;=\; K_d C. \tag{71}$$

The governing equation is given by

$$\frac{\partial}{\partial t}\varphi C + \nabla\cdot\big(\boldsymbol{q}C - \varphi D\nabla C\big) \;=\; -\frac{\partial S}{\partial t}. \tag{72}$$

If $C(x,\,t;\,\boldsymbol{q},\,D)$ is the solution to the case with no retardation (i.e. $K_d = 0$), then $C(x,\,t;\,\boldsymbol{q}/R,\,D/R)$ is the solution with retardation ($K_d > 0$), with

$$R = 1 + \frac{1}{\varphi}K_d. \tag{73}$$

Thus propagation of a front is retarded by the retardation factor R.

In operator splitting form this equation becomes

$$\frac{\partial}{\partial t}\varphi C + \nabla\cdot\big(\boldsymbol{q}C - \varphi D\nabla C\big) \;=\; 0, \tag{74}$$

and

$$\frac{d}{dt}\varphi C \;=\; -\frac{dS}{dt}. \tag{75}$$

The solution to the latter equation is given by

$$\varphi C^{t+\Delta t} - \varphi C^* \;=\; -\big(S^{t+\Delta t} - S^t\big), \tag{76}$$

where C^* is the solution to the nonreactive transport equation. Using Eqn.(**??**), this result can be written as

$$C^{t+\Delta t} \;=\; \frac{1}{R}C^* + \left(1 - \frac{1}{R}\right)C^t. \tag{77}$$

Thus for $R = 1$, $C^{t+\Delta t} = C^*$ and the solution advances unretarded.

4 Installation

4.1 MacOSX

This installation requires compiler versions 4.4.2 for gfortran, gcc, and g++. The compiler g++ is needed for installing SAMRAI. Define environment variables: `MPI_HOME`, `PKGS`, `HYPRE_INSTALL_DIR`

4.1.1 OpenMPI

```
setenv F90 gfortran
setenv F77 gfortran
setenv FC gfortran
setenv CC gcc-4
setenv CXX g++-4

./configure \
--prefix=$PKGS/openmpi/openmpi-1.4.1-gcc-4.4.2-gfortran \
--disable-debug \
--enable-static \
--disable-shared

make
make install
```

4.1.2 Hypre

```
./configure \
--with-MPI \
--enable-debug \
--disable-opt \
--prefix=${HYPRE_INSTALL_DIR} \
--with-MPI-include=${MPI_HOME}/include \
--with-MPI-libs=openmpi \
--with-MPI-lib-dirs=${MPI_HOME}/lib \
CC=mpicc \
CXX=mpicxx \
F77=mpif90

make
make install
```

4.1.3 PETSc

The PETSc `./configure` script can be used not only to install PETSc but also HDF5, Parmetis, and various direct solvers such as Mumps and Trilinos solvers. To use PETSc with PFLOTRAN the PETSc developer version petsc-dev is necessary which can be downloaded from the PETSc web page (http://www.mcs.anl.gov/petsc/petsc-as/developers/index.html). The recommended way to do this is to use the version control program Mercurial (http://mercurial.selenic.com/wiki).

Define environment variables: `PETSC_DIR` and `PETSC_ARCH` giving the location of the petsc-dev source and the architecture.

4.1.3.1 MacOSX 10.6.6 (SnowLeopard)

```
./config/configure.py \
--with-mpi-dir=$MPI_HOME \
--download-openmpi=1 \
--with-debugging=0 \
--with-shared-libraries=0 \
--download-mumps=1 \
--download-parmetis=1 \
--download-scalapack=1 \
--download-blacs=1 \
--download-ml=1 \
--download-hdf5=1

make all
make test
```

4.1.3.2 MacOSX 10.6.4 (Snow Leopard)

1. Obtain C compiler gcc4.5 or later for snow leopard (e.g. from http://hpc.sourceforge.net). Install gcc under `/opt/local` or `/usr/local`. You don't have to get gcc4.5 or 4.6, but you have to have a verison not older than gcc4.3 for some features in pflotran. Both gcc4.5 and gcc4.6 come with gfortran.

2. Install openmpi 1.5.1 from www.open-mpi.org. Configure using the command:

   ```
   ./configure CC=gcc CXX=g++ FC=gfortran --enable-mpi-f77=no
   --prefix=/opt/local
   ```

3. Download petsc-dev and configure using

   ```
   ./configure --download-hdf5=1 --with-cc=mpicc --with-cxx=mpicxx
   --with-fc=mpif90,
   ```

then compile with `make all`.

4. Compile pflotran using

   `make pflotran`

4.1.4 ORNL's Jaguar XT4/5

```
./config/configure.py PETSC_ARCH=cray-xt4-pgi_fast \
--configModules=PETSc.Configure \
--optionsModule=PETSc.compilerOptions \
--known-level1-dcache-size=65536 \
--known-level1-dcache-linesize=64 \
--known-level1-dcache-assoc=2 \
--known-memcmp-ok=1 \
--known-sizeof-char=1 \
--known-sizeof-void-p=8 \
--known-sizeof-short=2 \
--known-sizeof-int=4
--known-sizeof-long=8 \
--known-sizeof-long-long=8 \
--known-sizeof-float=4 \
--known-sizeof-double=8 \
--known-sizeof-size_t=8 \
--known-bits-per-byte=8 \
--known-sizeof-MPI_Comm=4 \
--known-sizeof-MPI_Fint=4 \
--known-mpi-long-double=0 \
--with-batch=1 \
--with-shared-libraries=0 \
--with-dynamic=0 \
--with-cc=cc \
--with-cxx=CC \
--with-fc=ftn \
--COPTFLAGS="-tp barcelona-64 -fastsse -Mipa=fast" \
--CXXOPTFLAGS="-tp barcelona-64 -fastsse -Mipa=fast" \
--FOPTFLAGS="-tp barcelona-64 -fastsse" \
--with-debugging=0 \
--with-blas-lib=sci \
--with-lapack-lib=sci \
--with-x=0 \
--with-mpi-dir=$MPICH_DIR \
--download-hypre=1 \
--download-parmetis=1 \
--with-hdf5=1 \
--with-hdf5-dir=$HDF5_DIR \
--known-mpi-shared=0
```

The user will need to load the following HDF5 module beforehand: `module load hdf5-parallel`.

4.2 Building PFLOTRAN

The source code for PFLOTRAN can be downloaded from the LANL web site using Mercurial: `hg clone https://software.lanl.gov/pflotran/hg/pflotran`. PFLOTRAN is compiled with the command:

```
make [chuan_co2=1 coll=1 temp=1 dasvyat=1] pflotran
```

Options [] are:

`chuan_co2`:	–enable two-phase supercritical CO_2 mode
`coll`:	–enable colloid-facilitated transport option
`temp`:	–enable temperature dependent log K capability
`dasvyat`:	–enable MFD full permeability tensor capability

4.3 Running PFLOTRAN

PFLOTRAN can be run from the command line as

```
mpirun -np 10 pflotran [options]
```

A number of command line options are available:

`-pflotranin` <input file name>	specify an input file [Default: `pflotran.in`]
`-screen_output off`	turn off screen output
`-file_output` <output file name>	specify an output file [Default: `pflotran.out`]
`-multisimulation`	run multiple input files in one run
`-stochastic`	Monte Carlo multiple realization run
`-log_summary`	print out run performance (PETSc option)

4.4 Building SAMRAI Version 2.4.4

To build SAMRAI follow the instructions listed below.

```
mkdir SAMRAI-v2.4.4

cd SAMRAI-v2.4.4

hg clone http://software.lanl.gov/pflotran/hg/samrai SAMRAI

mkdir samrai-objs in directory SAMRAI-v2.4.4

cd samrai-objs
```

Define the environment variables `SAMRAI_INSTALL_DIR`, `MPI_HOME`, `HDF5_HOME`, `PETSC_DIR` to be the top level directories of the appropriate packages

With MPICH:

```
../SAMRAI/configure --prefix=${SAMRAI_INSTALL_DIR} \
--with-CC=mpicc \
--with-CXX=mpicxx \
--with-F77=mpif90 \
--with-mpi \
--with-mpi-include=${MPI_HOME}/include \
--with-mpi-lib-dirs=${MPI_HOME}/lib \
--with-MPICC=mpicc \
--with-x \
--with-hdf5=${HDF5_HOME} \
--with-hypre=${HYPRE_HOME} \
--with-petsc=${PETSC_DIR} \
--with-blaslapack \
--enable-opt \
 --enable-debug \
--enable-char \
--enable-bool \
CXXFLAGS="-DMPICH_IGNORE_CXX_SEEK -DMPICH_SKIP_MPICXX" \
CPPFLAGS="-DMPICH_IGNORE_CXX_SEEK -DMPICH_SKIP_MPICXX"
```

With OPENMPI:

```
../${SAMRAI_SRC_DIR}/configure \
--prefix=${SAMRAI_INSTALL_DIR} \
--with-CC=mpicc \
--with-CXX=mpicxx \
--with-F77=mpif90 \
--with-mpi \
--with-mpi-include=${MPI_HOME}/include
--with-mpi-lib-dirs=${MPI_HOME}/lib
--with-MPICC=mpicc \
--with-x \
--with-hdf5=${HDF5_INSTALL_DIR} \
--with-hypre=${HYPRE_INSTALL_DIR} \
--with-petsc=${PETSC_DIR} \
--with-blaslapack \
--disable-opt \
--enable-debug \
--enable-char \
--enable-bool \
CXXFLAGS="-DOMPI_IGNORE_CXX_SEEK -DOMPI_SKIP_MPICXX \
  -I${PETSC_DIR}/${PETSC_ARCH}/include -I${PETSC_DIR}/include" \
CPPFLAGS="-DOMPI_IGNORE_CXX_SEEK -DOMPI_SKIP_MPICXX"
```

```
make
```

```
make install
```

Define the environment variable `SAMRAI` to point to `SAMRAI_INSTALL_DIR` (for example in your .cshrc)

Instructions for SAMR utils package

```
hg clone http://software.lanl.gov/pflotran/hg/samrutils
```

```
cd samrutils
```

Define the environment variable `AMRUTILITIES_HOME` to be the directory where you want the libraries and headers installed, it can point to the top level src dir if you like.

```
make prefix=${AMRUTILITIES_HOME} lib3d
```

Instructions for SAMR solvers package

```
hg clone http://software.lanl.gov/pflotran/hg/samrsolvers
```

```
cd samrsolvers
```

Define the environment variable `SAMRSOLVERS_HOME` to be the directory where you want the libraries and headers installed, it can point to the top level src dir if you like.

```
make prefix=${SAMRSOLVERS_HOME} lib3d
```

Assumptions:

1. External packages needed by PFLOTRAN are built: petsc-dev, hypre, hdf5, mpich2/openmpi/some mpi
2. External packages needed by PFLOTRAN AMR interface are built: SAMRAI, SAMRUtils, SAMRSolvers

Define the environment variables:

SAMRAI	–points to where SAMRAI is installed
AMRUTILITIES_HOME	–points to where SAMRUtils is installed
SAMRSOLVERS_HOME	–points to where SAMRSolvers is installed

Instructions for building `SAMRAIDriver`:

```
cd pflotran/src/pflotran
make samr_hdf5=1 pflotranamr

cd samr/src
make
```

At this point there should exist an executable named SAMRAIDriver.

4.4.1 Running PFLOTRAN/SAMRAI on Jaguar

Add to the `.tcshrc` (or equivalent) file:

```
umask 0002
set LD_LIBRARY_PATH=/usr/lib64:${LD_LIBRARY_PATH}
setenv ARCH cray-xt5
setenv COMPILER gcc-4.4.2
setenv OPT opt
#setenv OPT debug
setenv PROJ_DIR /tmp/proj/csc025/csc025geo3
setenv PKGS ${PROJ_DIR}/packages
setenv PKG_POSTFIX ${ARCH}-${COMPILER}-${OPT}
setenv PETSC_ARCH ${PKG_POSTFIX}
setenv PETSC_VERSION petsc-dev
setenv PETSC_DIR ${PKGS}/petsc/${PETSC_VERSION}-${OPT}
setenv SAMRAI ${PKGS}/samrai/SAMRAI-v2.4.4/${PETSC_VERSION}-${PKG_POSTFIX}
setenv AMRUTILITIES_HOME ${PKGS}/samrutils/${PKG_POSTFIX}
setenv SAMRSOLVERS_HOME ${PKGS}/samrsolvers/${PKG_POSTFIX}
setenv PCH ${SAMRSOLVERS_HOME}

setenv MPICH_UNEX_BUFFER_SIZE 400M
setenv MPICH_PTL_OTHER_EVENTS 4096
setenv MPICH_PTL_SEND_CREDITS -1
setenv MPICH_MSGS_PER_PROC 60000
```

Load the following modules:

```
module swap PrgEnv-pgi PrgEnv-gnu

module load mercurial
module load hdf5-parallel/1.8.3.1
module load hypre/2.4.0b
module load szip/2.1
module load totalview/8.6.0-1
```

The user will need to set: PETSC_DIR, PETSC_ARCH, AMRUTILITIES_HOME, SAMRSOLVERS_HOME to be defined as above. Also, note the swap to gnu.

```
cd pflotran
hg pull -u

cd src/pflotran
make hdf5=1 jaguar=1 pflotranamr
```

```
cd samr/src
make hdf5=1 jaguar=1
```

At this point you should have an executable `SAMRAIDriver`.

5 Creating the Input File: PFLOTRAN Keywords

The PFLOTRAN input file construction is based on keywords. Lines beginning with a colon (:) are treated as comments. Each entry to the input file must begin in the first column. Keywords `SKIP` and `NOSKIP` are used to skip over sections of the input file. Blank lines may occur in input file. Alternate keyword spelling is indicated in round brackets ( ). Input options are indicated in square brackets [ ], as well as default values. Curly brackets { } indicate the result of invoking the corresponding keyword. Always refer to source code when in doubt!

Initial and boundary conditions and material properties are assigned to spatial regions using a novel *coupler* approach. In this approach, initial and boundary conditions (keyword CONDITION) are assigned to regions (keyword REGION) using keywords INITIAL_CONDITION and BOUNDARY_CONDITION. Material properties (keyword MATERIAL) are assigned to regions using the keyword STRATIGRAPHY.

Keyword	Description
BOUNDARY_CONDITION	
BREAKTHROUGH	
BRINE (BRIN)	
CHECKPOINT	
CHEMISTRY	
COMPUTE_STATISTICS	
CONSTRAINT	transport (optional)
DATASET	
DEBUG	
FLOW_CONDITION	
FLUID_PROPERTY	
GRID	(required)
INITIAL_CONDITION	
LINEAR_SOLVER	
MATERIAL_PROPERTY	
MODE	
NEWTON_SOLVER	
NUMERICAL_JACOBIAN	
OBSERVATION	
ORIG, ORIGIN	
OUTPUT	
OVERWRITE_RESTART_TRANSPORT	
PROC	(optional)

REGION	
RESTART	
SATURATION_FUNCTION	
SOURCE_SINK	
STRATIGRAPHY (STRATA)	
TIME	
TIMESTEPPER	
TRANSPORT_CONDITION	
UNIFORM_VELOCITY	
USE_TOUCH_OPTIONS	
VELOCITY_DATASET	(optional)
WALLCLOCK_STOP	

Conventions and Notation: Keywords are in boldface with optional modifying keywords in square brackets [. . .], and user entries in typewriter font.

Unless otherwise specified, units in the input file are assumed to be as listed in Table **??**.

Table 2: Units

Quantity	Units
Pressure:	Pascal [Pa] (absolute)
Temperature:	Celcius [C]
Distance:	meter [m]
Volume:	meter^3 [m^3]
Time:	second [s]
Velocity:	meter/second [m/s]
Concentration:	molarity [M] or molality [m] (see MOLAL keyword)
Enthalpy:	kilojoule/mole [kJ/mol]
Mass:	kilogram [kg]
Rate:	mass/second [kg/s] or volume/second [m^3/s]
Surface Site Density:	mole/meter^3 [mol/m^3]

5.1 Example Input File

:colon denotes a comment line :Description: 3D infiltration problem with calcite dissolution

```
: == debugging =================================================================
:DEBUG
:MATVIEW_JACOBIAN
:VECVIEW_RESIDUAL
:VECVIEW_SOLUTION
:/

: == mode ======================================================================
MODE RICHARDS

: == chemistry =================================================================
CHEMISTRY
OPERATOR_SPLIT
PRIMARY_SPECIES
Ca++
H+
CO2(aq)
Tracer
/
SECONDARY_SPECIES
OH-
HCO3-
CO3--
CaHCO3+
CaCO3(aq)
/
GAS_SPECIES
CO2(g)
/
MINERALS
Calcite
/
:
MINERAL_KINETICS
Calcite
RATE_CONSTANT 1.e-8 ! [mol/cm^3/s]
/
/
:
DATABASE /Users/lichtner/flotran/database/hanford.dat
LOG_FORMULATION
ACTIVITY_COEFFICIENTS !NEWTON_ITERATION
MOLAL
OUTPUT
All
/
/

: == reference variables =======================================================
REFERENCE_POROSITY 0.25d0


: == time stepping =============================================================
TIMESTEPPER
TS_ACCELERATION 8
MAX_STEPS 100000
/
```

```
: == discretization ==========================================================
GRID
:TYPE amr ! amr grid
TYPE structured
NXYZ 6 6 6
DXYZ
1.
1.
1.
/
/

: == flow solvers =============================================================
NEWTON_SOLVER FLOW
PRECONDITIONER_MATRIX_TYPE AIJ
RTOL 1.d-8
ATOL 1.d-8
STOL 1.d-30
ITOL_UPDATE 1.d0
:NO_INFINITY_NORM
:NO_PRINT_CONVERGENCE
:PRINT_DETAILED_CONVERGENCE
/

LINEAR_SOLVER FLOW
:KSP_TYPE PREONLY
:PC_TYPE LU
:KSP_TYPE FGMRES !samrai
:PC_TYPE SHELL !samrai
/

: == transport solvers ========================================================
NEWTON_SOLVER TRANSPORT
PRECONDITIONER_MATRIX_TYPE AIJ
RTOL 1.d-12
ATOL 1.d-12
STOL 1.d-30
:NO_INFINITY_NORM
:NO_PRINT_CONVERGENCE
:PRINT_DETAILED_CONVERGENCE
/

LINEAR_SOLVER TRANSPORT
:PC_TYPE LU
:KSP_TYPE PREONLY
:KSP_TYPE FGMRES ! samrai
:PC_TYPE SHELL !samrai
/

: == fluid properties =========================================================
FLUID_PROPERTY
DIFFUSION_COEFFICIENT 1.d-9
/

: == material properties ======================================================
MATERIAL_PROPERTY HD
ID 1
SATURATION_FUNCTION HD
```

```
POROSITY 0.262
TORTUOSITY 1.0
PERMEABILITY
PERM_ISO 5.43d-13
/
/

: == saturation / permeability functions ======================================
SATURATION_FUNCTION HD
SATURATION_FUNCTION_TYPE VAN_GENUCHTEN
RESIDUAL_SATURATION 0.115
LAMBDA 0.286
ALPHA 1.9401d-4
/

: == output ====================================================================
OUTPUT
:PERIODIC TIMESTEP 1
PERIODIC TIME 0.1 y
FORMAT HDF5
FORMAT TECPLOT BLOCK
VELOCITIES
/

: == times =====================================================================
TIME
FINAL_TIME 1.d0 y
INITIAL 1.e-6 y
MAXIMUM_TIMESTEP_SIZE 1.e-2 y
/

: == regions ===================================================================
REGION all
COORDINATES
0.d0 0.d0 0.d0
6.d0 6.d0 6.d0
/
/

REGION Top
FACE TOP
COORDINATES
0.d0 0.d0 6.d0
6.d0 6.d0 6.d0
/
/

REGION Inlet
FACE TOP
COORDINATES
2.d0 2.d0 6.d0
4.d0 4.d0 6.d0
/
:BLOCK 3 4 3 4 6 6
/

REGION Bottom
FACE BOTTOM
```

```
COORDINATES
0.d0 0.d0 0.d0
6.d0 6.d0 0.d0
/
/

: == flow conditions ============================================================
FLOW_CONDITION Inlet
TYPE
FLUX neumann
/
FLUX 0.317098d-6 ! 10 m/y
/

FLOW_CONDITION Initial
TYPE
PRESSURE hydrostatic
/
DATUM 0.d0 0.d0 6.d0
PRESSURE 101325.d0
/

: == transport conditions =======================================================
TRANSPORT_CONDITION Inlet
TYPE dirichlet
CONSTRAINT_LIST
0.d0 Inlet
/
/

TRANSPORT_CONDITION Initial
TYPE dirichlet
CONSTRAINT_LIST
0.d0 Initial
/
/

TRANSPORT_CONDITION Outlet
TYPE zero_gradient
CONSTRAINT_LIST
0.d0 Initial
/
/

: == couplers ===================================================================
BOUNDARY_CONDITION Inlet
FLOW_CONDITION Inlet
TRANSPORT_CONDITION Inlet
REGION Inlet
/

BOUNDARY_CONDITION Outlet
FLOW_CONDITION Initial
TRANSPORT_CONDITION Outlet
REGION Bottom
/

INITIAL_CONDITION Initial
```

```
FLOW_CONDITION Initial
TRANSPORT_CONDITION Initial
REGION all
/

: == stratigraphy ==========================================================
STRATA
MATERIAL HD
REGION all
/

: == transport constraints =================================================
CONSTRAINT Initial
CONCENTRATIONS
Ca++ 1.d-4 M Calcite
H+ 8.0d0 pH
CO2(aq) 1.d-2 G CO2(g)
Tracer 1.d-8 T
/
MINERALS
Calcite 0.75 1.
/
/

CONSTRAINT Inlet
CONCENTRATIONS
Ca++ 1.d-6 T
H+ 3.0d0 pH
CO2(aq) 1.d-3 G CO2(g)
Tracer 1.d-0 T
/
/
```

5.2 Keyword: BOUNDARY_CONDITION

Description: The BOUNDARY_CONDITION keyword couples conditions specified under the FLOW_CONDITION and/or TRANSPORT_CONDITION keywords to a REGION in the problem domain. The use of this keyword enables the use/reuse of flow and transport conditions and regions within multiple boundary and initial conditions and source/sinks in the input deck.

Input:

BOUNDARY_CONDITION `boundary_condition_name`

  FLOW_CONDITION `flow_condition_name`

  TRANSPORT_CONDITION `transport_condition_name`

  REGION `region_name`

(., /, END)

Explanation:

Keyword	Description
BOUNDARY_CONDITION	Defines the beginning of a boundary condition entry and the name of the boundary condition.
FLOW_CONDITION	Defines the name of the flow condition to be linked to this boundary condition.
TRANSPORT_CONDITION	Defines the name of the transport condition to be linked to this boundary condition.
REGION	Defines the name of the region to which the conditions are linked.
END	Terminates the boundary condition entry.

Examples:

```
BOUNDARY_CONDITION river
FLOW_CONDITION river_stage
TRANSPORT_CONDITION river_chemistry
REGION river_bank
END

BOUNDARY_CONDITION recharge
```

```
FLOW_CONDITION infiltration_flux
TRANSPORT_CONDITION infiltration_chemistry
REGION ground_surface
END
```

5.3 Keyword: BRINE (BRIN)

BRINE, BRIN `Value m_nacl` [MOLAL, MASS, MOLE]

5.4 Keyword: CHECKPOINT

Description: Checkpoint files enable the restart of a simulation at any discrete point in simulation where a checkpoint file has been printed. When the CHECKPOINT card is included in the input deck, checkpoint files are printed every N time steps, where N is the checkpoint frequency, and at the end of the simulation, should the simulation finish or the be shut down properly mid-simulation using the WALL_CLOCK_STOP card. Checkpoint files are named `pflotran.chkN`, where N is the number of the timestep when the checkpoint file was printed. A file named `restart.chk` will also be written when PFLOTRAN properly terminates execution. One use this file to pick up from where the simulation stopped by increasing the final time.

Input:

CHECKPOINT `<checkpoint_frequency>`

Explanation:

Keyword	Description
CHECKPOINT	toggles on checkpointing
`checkpoint_frequency`	frequency at which checkpoint files are printed `<integer>`

Examples:

```
CHECKPOINT 1000
```

```
CHECKPOINT 5
```

5.5 Keyword: CHEMISTRY

Description: The **CHEMISTRY** keyword invokes the reactive transport mode and provides input for primary species, secondary species, minerals, gases, colloids and colloid-facilitated transport, and sorption including ion exchange and surface complexation. Mineral reactions are described through a kinetic rate law based on transition state theory and surface complexation reactions may involve equilibrium, kinetic (reversible or irreversible) or a multirate formulation.

Input:

CHEMISTRY

PRIMARY_SPECIES

`Name`

(., /, END)

SECONDARY_SPECIES

`Name`

(., /, END)

GAS_SPECIES

`Name`

(., /, END)

MINERALS

`Name`

(., /, END)

COLLOIDS

`Name` `Mobile_Fraction` [—]

(., /, END)

MINERAL_KINETICS

`Mineral Name`

RATE_CONSTANT `Value` [mol/cm^2/s]

(., /, END)

(., /, END)

SORPTION

SURFACE_COMPLEXATION_RXN

EQUILIBRIUM

MULTIRATE_KINETIC

KINETIC

COMPLEX_KINETICS

FORWARD_RATE_CONSTANT `Value` [mol/cm^3/s???]
BACKWARD_RATE_CONSTANT `Value` [mol/cm^3/s???]
SITE_FRACTION `Value`[—] (Continuation line '\')
RATE, RATES `Value` [mol/cm^3/s?] (Continuation line '\')
MULTIRATE_SCALE_FACTOR `Value` [—]
MINERAL `Mineral Name`
SITE `Name Site Density` [mol/m^3]
COMPLEXES
`Complex Name`
(., /, END)
(., /, END)
COLLOID `Name`
SITE `Name Site Density` [mol/m^3]
COMPLEXES
`Complex Name`
(., /, END)
(., /, END)
(., /, END)

ION_EXCHANGE_RXN
MINERAL `Mineral Name`
CEC `Value`[mol/m^3]
CATIONS
`Name`
(., /, END)
(., /, END)
(., /, END)
DISTRIBUTION_COEF (not implemented)
JUMPSTART_KINETIC_SORPTION
NO_CHECKPOINT_KINETIC_SORPTION
NO_RESTART_KINETIC_SORPTION

(., /, END)

DATABASE `Path/Database_Name`

LOG_FORMULATION

NO_CHECKPOINT_ACT_COEFS

ACTIVITY_COEFFICIENTS [**LAG, NEWTON, TIMESTEP, NEWTON_ITERATION**]

ACTIVITY_H2O, ACTIVITY_WATER

MOLAL, MOLALITY

NO_BDOT

UPDATE_POROSITY

UPDATE_TORTUOSITY

UPDATE_PERMEABILITY

UPDATE_MINERAL_SURFACE_AREA

MAX_DLNC (Default 5)

OUTPUT

MOLALITY

MOLARITY

All

`Species Name`

FREE_ION

pH

TOTAL_SORBED

TOTAL_SORBED_MOBILE

COLLOIDS

KD

(., /, END)

(., /, END)

Explanation:

Keyword	**Description**
Primary_Species	List of primary species that fully describe the chemical composition of the fluid. The set of primary species must form an independent set of species in terms of which all homogeneous aqueous equilibrium reactions can be expressed.
Secondary_Species	List of aqueous species in equilibrium with primary species.
Gas_Species	List of gas species.
...	

Examples:

5.6 Keyword: COMPUTE_STATISTICS

Description: COMPUTE_STATISTICS enables the calculation statistical analysis of flow velocities during a simulation. The average, maximum, minimum, and standard deviations velocities are computed.

Input:

COMPUTE_STATISTICS {compute_statistics = .true.}

Explanation:

Example:

```
COMPUTE_STATISTICS
```

5.7 Keyword: CONSTRAINT

Description: The keyword **CONSTRAINT** sets up fluid compositions based on various constraint conditions chosen by the user.

Input:

CONSTRAINT `constrain_name`

CONC, CONCENTRATIONS `Name, Concentration_Value, Constraint`
The variable `Constraint` is chosen from the following list:

F, FREE

T, TOTAL

TOTAL_SORB

P, PH

L, LOG

M, MINERAL, MNRL

G, GAS

SC, CONSTRAINT_SUPERCRIT_CO2

Z, CHG

(., /, END)

MNRL, MINERALS

`mineral_name, volume_fraction, surface_area`

(., /, END)

(., /, END)

Explanation:

Example:

```
CONSTRAINT initial
CONCENTRATIONS
H+        7.3              pH
O2(aq)    1.78132e-4        T
Al+++     1.e-9             M K-Feldspar
Ca++      1.20644e-3        M Calcite
Cu++      1.e-6             T
Fe++      1.e-9             M Ferrihydrite
Mg++      5.09772e-4        T
UO2++     2.34845e-7        T
K+        1.54789e-4        T
Na+       2.03498e-3        T
HCO3-     2.57305e-3        T
```

```
Cl-      6.97741e-4       T
F-       2.09491e-5       T
HPO4--   1.e-6            T
NO3-     4.69979e-3       T
SO4--    6.37961e-4       T
SiO2(aq) 5.36989e-4       T
Tracer   2.34845e-7       F
/
MINERALS
Quartz         0.35  1.
Calcite        0.    1.
Metatorbernite 0.    1.
/
/
```

5.8 Keyword: DATASET

Description: Dataset for permeability.

Input:

DATASET [permx, permy, permz] [permx_filename, permy_filename, permz_filename]

Input:

5.9 Keyword: DEBUG

DEBUG

PRINT_SOLUTION [VECVIEW_SOLUTION, VIEW_SOLUTION]

PRINT_RESIDUAL [VECVIEW_RESIDUAL,VIEW_RESIDUAL]

PRINT_JACOBIAN [MATVIEW_JACOBIAN, VIEW_JACOBIAN]

PRINT_JACOBIAN_NORM [NORM_JACOBIAN]

PRINT_COUPLERS [PRINT_COUPLER]

PRINT_JACOBIAN_DETAILED [MATVIEW_JACOBIAN_DETAILED, VIEW_JACOBIAN_DETAILED]

PRINT_NUMERICAL_DERIVATIVES] [VIEW_NUMERICAL_DERIVATIVES]

(.,/,END)

Explanation:

Examples:

5.10 Keyword: FLOW_CONDITION

Description: The **FLOW_CONDITION** keyword specifies scalar or vector data sets to be associated with a given boundary or initial condition. For instance, to specify a hydrostatic boundary condition, the use would specify a condition with a pressure associated with a point in space (i.e. datum) in space and a gradient, both vector quantities. Note that in the case of a hydrostatic boundary condition, the vertical gradient specified in the input deck must be zero in order to enable the hydrostatic pressure calculation. Otherwise, the specified vertical gradient overrides the hydrostatic pressure. Transient pressures, temperatures, concentrations, datums, gradients, etc. are specified using the **FILE** `filename` combination for the name of the data set.

Input:

FLOW_CONDITION `flow_condition_name`

UNITS `Value` (not currently supported)
`Value` is one of the following entries:

s, sec, min, hr, d, day, w, week, mo, month, y, yr (time)

mm, cm, m, met, meter, dm, km (length)

kg/s, kg/yr (rate)

Pa, KPa (pressure)

m/s, m/yr (velocity)

C, K (temperature)

M, mol/L (concentration)

KJ/mol (enthalpy)

CYCLIC

INTERPOLATION

step

linear

TYPE

PRESSURE `[dirichlet, hydrostatic, zero_gradient, conductance, seepage]`

RATE `[mass_rate, volumetric_rate, scaled_volumetric_rate]`: specifies an injection/extraction rate in mass [kg/s], volume [m^3/s], and a volumetric injection/extraction rate [m^3/s] that is scaled across a well screen, weighted as a function of the interfacial area and permeability of neighboring cells (in x, y).

FLUX `[dirichlet, neumann, mass_rate, hydrostatic, conductance, zero_gradient, production_well, seepage, volumetric, volumetric_rate, equilibrium]`

TEMPERATURE `[dirichlet, hydrostatic, zero_gradient]`

CONCENTRATION `[dirichlet, hydrostatic, zero_gradient]`

ENTHALPY (H) `[dirichlet, hydrostatic, zero_gradient]`

(., /, END)

TIME (not currently supported)

IPHASE `Value[integer]`

DATUM

`x y z`

FILE `file_name`

GRADIENT, GRAD

PRES, PRESS, PRESSURE

d_{dx} d_{dy} d_{dz}

FILE `file_name`

FLUX

TEMP, TEMPERATURE

CONC, CONCENTRATION

H, ENTHALPY

(., /, END)

TEMPERATURE, TEMP `<float>`

ENTHALPY, H `<float>`

PRESSURE, PRES, PRESS `<float>`

RATE `Value`

FLUX, VELOCITY, VEL `<float>`

CONC, CONCENTRATION `<float>`

CONDUCTANCE `<float>`

(., /, END)

Explanation:

Keyword	Description
FLOW/TRANSPORT_CONDITION	Initiates a condition entry and defines its name.
CYCLIC	Instructs PFLOTRAN to cycle the transient data set should the simulation time exceed the last time in the data set.
INTERPOLATION	Defines the method for interpolating between data set times.
DATUM	Location is space where prescribed scalar (e.g. pressure, temperature concentration, etc.) is defined.
TYPE	Specifies the type of condition.
PRESSURE	Specifies the type of pressure condition.
TEMPERATURE	Specifies the type of temperature condition.
CONCENTRATION	Specifies the type of concentration condition.
ENTHALPY	Specifies the type of enthalpy condition.
END	Terminates type entry.
GRADIENT	Gradient of the scalar field in 3D space.
PRESSURE	Pressure gradient in x-, y-, and z-directions.
TEMPERATURE	Temperature gradient in x-, y-, and z-directions.
CONCENTRATION	Concentration gradient in x-, y-, and z-directions.
ENTHALPY	Enthalpy gradient in x-, y-, and z-directions.
END	Terminates gradient entry.
PRESSURE	Absolute fluid pressure at the datum.
FLUX	Darcy velocity of fluid defining flux across a boundary.
TEMPERATURE	Temperature in °C at the datum.
CONCENTRATION	Solute concentration at the datum.
ENTHALPY	Enthalpy at the datum.
CONSTRAINT_LIST	Specifies a list of concentration constraints for solute transport.
END	Terminates the condition entry.

Examples:

```
FLOW_CONDITION initial
TYPE
```

```
PRESSURE hydrostatic
/
PRESSURE 1956741.84 ! 200 meter piezometric head (200*997.32*9.81)
/

FLOW_CONDITION source
TYPE
RATE volumetric_rate
/
RATE 2.77777778d-3  ! 10 m^3/hr
/

TRANSPORT_CONDITION initial
TYPE zero_gradient
CONSTRAINT_LIST
0.d0 initial
/
END

TRANSPORT_CONDITION source
TYPE dirichlet
CONSTRAINT_LIST
0.d0 well
/

FLOW_CONDITION East
TYPE
:PRESSURE seepage
PRESSURE conductance
/
CYCLIC
DATUM file ../../river_scope3.datum
GRADIENT
PRESSURE file ../../river_scope3.gradient
/
CONDUCTANCE 1.d-12
PRESSURE 101325.d0
/
```

5.11 Keyword: FLUID_PROPERTY

Description:

Input:

FLUID_PROPERTY

PHASE `<name>` (LIQUID_PHASE, GAS_PHASE) [Default: LIQUID_PHASE]

DIFFUSION_COEFFICIENT `<float>` [Default: 1×10^{-9} m^2/s]

(., /, END)

Explanation:

Example:

```
FLUID_PROPERTY
DIFFUSION_COEFFICIENT 1.d-9
/
```

5.12 Keyword: GRID **Required**

Description: this keyword defines the descritization scheme, the type of grid and resolution, and the geometry employed in the simulation.

GRID

Required Input Parameters:

TYPE <type> <symmetry>:

Grid type (structured, structured_mimetic, unstructured, amr)
Symmetry type (cartesian [default], cylindrical, spherical

NXYZ <# # #>: # of grid cells in x, y, z directions (structured only)

FILE <filename>: Name of file containing grid information (unstructured only)

BOUNDS: Specifies bounds of structured cartesian grid (see examples below)

<x_min, x_max>

<y_min, y_max>

<z_min, z_max>

(.,/,END)

DXYZ: Specifies grid spacing of structured cartesian grid (see examples below)

<dx>

<dy>

<dz>

(.,/,END)

Optional Input Parameters:

GRAVITY <# # #>: Specifies gravity vector [Default: 0 0 −9.8068 m/s^2]

ORIGIN <# # #>: Coordinate of grid origin [Default: 0 0 0]

INVERT_Z: Inverts the z-axis [Default: positive z points downward]

Examples:

```
GRID
TYPE structured cylindrical
NXYZ 512  1  32
DXYZ
2.d0
1.d0
2.d0
END
```

```
BOUNDS
0. 1024.
0.    1.
0.   64.
END
```

By using the BOUNDS keyword, the model domain is specified in a grid-independent fashion and, as a result, the grid spacing may be changed by modifying the keyword NXYZ only.

5.13 Keyword: INITIAL_CONDITION

Description: Condition coupler between regions and flow and transport conditions.

Input:

INITIAL_CONDITION `[Name]`

> **REGION** `region_name`
>
> **FLOW_CONDITION** `condition_name`
>
> **TRANSPORT_CONDITION** `condition_name`

(., /, END)

Explanation:

Example:

```
:=========== condition couplers ==============
: initial condition
INITIAL_CONDITION
FLOW_CONDITION gradient-north
TRANSPORT_CONDITION Initial
REGION all
END
```

5.14 Keyword: LINEAR_SOLVER

Description:

Input:

LINEAR_SOLVER [**TRAN, TRANSPORT / FLOW**]

SOLVER_TYPE (SOLVER, KRYLOV_TYPE, KRYLOV, KSP, KSP_TYPE)

NONE (PREONLY)

GMRES

FGMRES

BCGS (BICGSTAB, BI-CGSTAB)

IBCGS (IBICGSTAB, IBI-CGSTAB)

RICHARDSON

CG

PRECONDITIONER_TYPE (PRECONDITIONER, PC, PC_TYPE)

NONE (PCNONE)

ILU (PCILU)

LU (PCLU)

BJACOBI (BLOCK_JACOBI)

ASM (ADDITIVE_SCHWARTZ)

PCASM

HYPRE

SHELL

HYPRE_OPTIONS TYPE [`pilut, parasails, boomeramg, euclid`]

BOOMERAMG_CYCLE_TYPE / [`V, W`]

BOOMERAMG_MAX_LEVELS `Value`

BOOMERAMG_MAX_ITER `Value`

BOOMERAMG_TOL `Value`

BOOMERAMG_TRUNCFACTOR `Value`

BOOMERAMG_AGG_NL `Value`

BOOMERAMG_AGG_NUM_PATHS `Value`

BOOMERAMG_STRONG_THRESHOLD `Value`

BOOMERAMG_GRID_SWEEPS_ALL `Value`

BOOMERAMG_GRID_SWEEPS_DOWN `Value`

BOOMERAMG_GRID_SWEEPS_UP `Value`

BOOMERAMG_GRID_SWEEPS_COARSE `Value`

BOOMERAMG_RELAX_TYPE_ALL `Value`

BOOMERAMG_RELAX_TYPE_DOWN `Value`

BOOMERAMG_RELAX_TYPE_UP `Value`

BOOMERAMG_RELAX_TYPE_COARSE `Value`
BOOMERAMG_RELAX_WEIGHT_ALL `Value`
BOOMERAMG_RELAX_WEIGHT_LEVEL `Value`
BOOMERAMG_OUTER_RELAX_WEIGHT_ALL `Value`
BOOMERAMG_OUTER_RELAX_WEIGHT_LEVEL `Value`
BOOMERAMG_NO_CF `Value`
BOOMERAMG_MEASURE_TYPE `Value`
BOOMERAMG_COARSEN_TYPE `Value`
BOOMERAMG_INTERPOLATION_TYPE, BOOMERAMG_INTERP_TYPE `Value`
BOOMERAMG_NODAL_COARSEN `Value`
BOOMERAMG_NODAL_RELAXATION `Value`

ATOL `Value`

RTOL `Value`

DTOL `Value`

MAXIT `Value`

(., /, END)

Explanation:

Example:

5.15 Keyword: MATERIAL_PROPERTY

Description: Specifies material properties to be associated with a region in the problem domain.

Input:

MATERIAL_PROPERTY `Name`

- **ID** `float`
- **SATURATION_FUNCTION** `Name`
- **ROCK_DENSITY** `float` [kg/m^3]
- **SPECIFIC_HEAT** `float` [kJ/(kg K)]
- **LONGITUDINAL_DISPERSIVITY** `float` [m]
- **TRANSVERSE_DISPERSIVITY** (not implemented) [m]
- **THERMAL_CONDUCTIVITY_DRY** `float` [W/(m K)]
- **THERMAL_CONDUCTIVITY_WET** `float` [W/(m K)]
- **PORE_COMPRESSIBILITY** `float` (not implemented) [bar^{-1}]
- **THERMAL_EXPANSITIVITY** `float` (not implemented) [C^{-1}]
- **POROSITY** `float` [—], `porosity_filename`
- **TORTUOSITY** `float` [—]
- **PERMEABILITY**
  - **ISOTROPIC** Toggles on isotropy
  - **ANISOTROPIC** Toggles on anisotropy
  - **VERTICAL_ANISOTROPY_RATIO** `float`
  - **PERM_X** `<float>` Diagonal permeability k_{xx} [m^2]
  - **PERM_Y** `<float>` Diagonal permeability k_{yy} [m^2]
  - **PERM_Z** `<float>` Diagonal permeability k_{zz} [m^2]
  - **PERM_ISO** `<float>` Isotropic permeability values [m^2]
  - **PERM_XY** `<float>` Off-diagonal permeability k_{xy} for use with MFD (mimetic_unstructured grid) [m^2]
  - **PERM_XZ** `<float>` [m^2] Off-diagonal permeability k_{xz}
  - **PERM_YZ** `<float>` [m^2] Off-diagonal permeability k_{yz}
- **(., /, END)**
- **PERMEABILITY_POWER** `float`
- **TORTUOSITY_POWER** `float`
- **MINERAL_SURFACE_AREA_POWER**
  - **VOLUME_FRACTION** `float` Volume fraction power in mineral surface area
  - **POROSITY** `float` Porosity power in mineral surface area

(., /, END)

RANDOM_DATASET `permeability_filename`

(., /, END)

Explanation:

Example:

```
MATERIAL_PROPERTY  Hanford
ID 1
SATURATION_FUNCTION sf1
POROSITY 0.332
TORTUOSITY 1.
PERMEABILITY
PERM_X 1.d-12
PERM_Y 1.d-12
PERM_Z 1.d-12
/
/
```

5.16 Keyword: MAX_CHANGE

Description:

Input:

MAX_CHANGE DPMAX DTMAX DSMAX DCMAX

Explanation:

Example:

```
:          dpmax dtmax dsmax dcmax
MAX_CHANGE 5.d4    5.  0.02   0.05
/
/
```

5.17 Keyword: MODE

Description: determines the flow mode: Richards (variably saturated porous media); MPH, MPHASE, FLASH2 (CO_2 + H_2O); THC (Thermal-Hydrologic-Chemical, in progress); IMMIS, THS (Immisible).

MODE <option>

Option	Description
RICHARDS	Single-phase, isothermal, variable saturated groundwater flow using Richards equation
MPHASE (MPH)	Multiphase supercritical CO_2-brine-energy based on variable switching for phase changes
FLASH2	Multiphase supercritical CO_2-brine-energy based on the flash method for phase changes with a persistent set of unknowns—required for AMR
THC	Thermo-Hydro-Chemical coupled groundwater flow, thermal and solute transport
IMMIS (IMS, THS)	Immissible CO_2-water-energy

Example:

```
MODE THC
```

5.18 Keyword: NEWTON_SOLVER

Description:

Input:

NEWTON_SOLVER

> **TRAN, TRANSPORT (tran_solver) / DEFAULT (flow_solver)**
>
> **INEXACT_NEWTON**
>
> **NO_PRINT_CONVERGENCE**
>
> **NO_INF_NORM (NO_INFINITY_NORM)**
>
> **NO_FORCE_ITERATION**
>
> **PRINT_DETAILED_CONVERGENCE**
>
> **ATOL**
>
> **RTOL**
>
> **STOL**
>
> **DTOL**
>
> **ITOL (INF_TOL, ITOL_RES, INF_TOL_RES)**
>
> **ITOL_UPDATE (INF_TOL_UPDATE)**
>
> **MAXIT**
>
> **MAXF**

(., /, END)

Explanation:

Example:

5.19 Keyword: NUMERICAL_JACOBIAN

NUMERICAL_JACOBIAN {numerical_derivatives = .true.}

5.20 Keyword: OBSERVATION

Description: The OBSERVATION card specifies a location (REGION) at which flow and transport results (e.g. pressure, saturation, flow velocities, solute concentrations, etc.) will be monitored in the output. The user must specify either a region or boundary condition to which the observation object is linked. The velocity keyword toggles on the printing of velocities at a point in space.

Input:

OBSERVATION

> **BOUNDARY_CONDITION** `boundary condition name`
>
> **REGION** `region name`
>
> **VELOCITY**
>
> **AT_CELL_CENTER**

(., /, END)

Explanation:

Keyword OBSERVATION initiates an observation point entry.

Keyword REGION (optional) defines the name of the region (usually a point in space) to which the observation point is linked.

Keyword BOUNDARY_CONDITION (optional) specifies the name of a boundary condition to which the observation point is tied (e.g. to monitor fluxes across a boundary face).

Keyword VELOCITY (optional) toggles on the printing of Darcy velocities at the observation point.

Examples:

```
OBSERVATION
REGION well1
VELOCITY
END

OBSERVATION
BOUNDARY_CONDITION river
END
```

5.21 Keyword: ORIGIN (ORIG)

ORIGIN (ORIG) X_DIRECTION Y_DIRECTION Z_DIRECTION

5.22 Keyword: OUTPUT

Description: The **OUTPUT** keyword controls formatting and time of output.

Input:

OUTPUT

- **TIMES** `Unit (s, h, y) <float>`
- **SCREEN OFF** suppress screen output
- **SCREEN PERIODIC** `<integer>`: print to screen every <integer> time steps.
- **PERIODIC TIME** `<float> Unit`
- **PERIODIC TIMESTEP** `<float> Unit`
- **PERIODIC_OBSERVATION TIME** `<float> <unit>`: output the results at observation points at times specified by the actual time
- **PERIODIC_OBSERVATION TIMESTEP** `<integer> <unit>`: output the results at observation points at times specified by time steps
- **NO_PRINT_INITIAL:** the initial state of the system will not be printed to the output file if this card is activated
- **NO_PRINT_FINAL:** the final state of the system will not be printed to the output file if this card is activated
- **FORMAT** <file format>: specify the snapshot file type. Options available are TECPLOT BLOCK, TECPLOT POINT, HDF5, MAD, VTK
- **PERMEABILITY**
- **POROSITY**
- **FLUXES**
- **VELOCITIES**
- **MASS_BALANCE:** output the mass balance of the system if this card is activated. It include global mass balance as well as fluxes at all boundaries for water and chemical species specified for output in the CHEMISTRY card.

(., /, END)

Explanation:

OUTPUT:	keyword to control output.
TIMES:	list of output times.
SCREEN OFF:	turns off screen output
SCREEN PERIODIC:	controls screen output frequency.
PERIODIC TIME:	controls frequency of output times.
PERIODIC TIMESTEP:	controls frequency of output time steps.
PERIODIC_OBSERVATION TIME:	frequency of output time.
PERIODIC_OBSERVATION TIMESTEP:	frequency of output time step.
NO_FINAL, NO_PRINT_FINAL:	
FORMAT TECPLOT POINT:	Tecplot POINT output, valid for 1D and 2D problems.
FORMAT TECPLOT BLOCK:	Tecplot BLOCK output.
FORMAT HDF5:	HDF5 output format written to a .h5 file which can be read by Visit.
FORMAT MAD:	MAD (Method of Anchored Distributions) format.
FORMAT VTK:	VTK format.
UNIT:	time units of seconds (s), days (d), and years (y).
PERMEABILITY:	
POROSITY:	
FLUXES:	
VELOCITIES:	keyword to output velocities.
MASS_BALANCE:	keyword to output global mass balances and boundary fluxes.

Examples:

```
OUTPUT
:SCREEN PERIODIC 10
:PERIODIC TIME 10 h
PERIODIC_OBSERVATION TIMESTEP 1
:times h 1.
:PERIODIC_OBSERVATION TIME 50 h
FORMAT TECPLOT POINT
FORMAT HDF5
VELOCITIES
MASS_BALANCE
/
```

5.23 Keyword: OVERWRITE_RESTART_TRANSPORT

OVERWRITE_RESTART_TRANSPORT {overwrite_restart_transport = .true.}

5.24 Keyword: REGION

Description: The **REGION** keyword defines a set of grid cells encompassed by a volume or intersected by a plane or point, or a list of grid cell ids. The **REGION** name can then be used to link this set of grid cells to material properties, strata, boundary and initial conditions, source sinks, observation points, etc. Although a region may be defined though the use of (I, J, K) indices using the **BLOCK** keyword, the user is encouraged to define regions either through **COORDINATES** or lists read in from an HDF5 file in order to minimize the dependence of the input file on grid resolution. In the case of the **FILE** keyword, a list of grid cell ids is read from an HDF5 file where the `region_name` defines the HDF5 data set. It should be noted that given a region defined by a plane or point shared by two grid cells (e.g. a plane defining the surface between two grid cells), **PFLOTRAN** will select the upwind cell(s) as the region.

Input:

REGION `region_name`

> **FILE** `file_name`
>
> **LIST** (to be implemented)
>
> **FACE** `face_name`
>
> **BLOCK** `i1 i2 j1 j2 k1 k2`
>
> **COORDINATE** `x y z`
>
> **COORDINATES**
>
> > `x1 y1 z1`
> >
> > `x2 y2 z2`
>
> **(., /, END)**

(., /, END)

Explanation:

Keyword REGION begins a region entry with name `region_name`.

Keyword BLOCK defines a volumetric, planar, or point region through IJK indices: i1 i2 j1 j2 k1 k2.

Keyword COORDINATE defines a point region through coordinates in 3D space.

Keyword COORDINATES Defines a volumetric, planar, or point region between two points in space.

Keyword FILE Defines an HDF5 file within which a dataset named region_name contains a list of grid cells corresponding to a region.

Keyword FACE Defines the face of the grid cell to which boundary conditions are connected where face_name is one of WEST, EAST, NORTH, SOUTH, BOTTOM, TOP (structured grids only).

Keyword END Ends the region entry (can be one of . END).

Examples:

```
REGION source_zone
BLOCK 3 5 15 16 2 3
END

REGION source_zone
BLOCK
3 5 15 16 2 3
END

REGION west_boundary
BLOCK 1 1 1 30 1 50
FACE WEST
END

REGION source_zone
COORDINATES
50. 10. 10.
60. 15. 15.
/
END

REGION river_boundary
FILE ./regions.h5
FACE EAST
END

REGION well
COORDINATE 50. 10. 10.
END

REGION well
COORDINATE
50. 10. 10.
END

REGION west_boundary
COORDINATES
0. 0. 0.
```

```
0. 10. 10.
/
FACE WEST
END
```

5.25 Keyword: RESTART

Description

The RESTART card defines a checkpoint file from which the current simulation should be restarted. If a time is specified after the file name, the initial simulation time is set to that time.

Input:

RESTART <restart_file_name> <restart_time> <time_units>

Explanation:

Keyword RESTART defines the checkpoint filename to be read in to restart a simulation at the specified time.

Examples:

```
RESTART restart.chk 0. y

RESTART restart.chk
```

5.26 Keyword: SATURATION_FUNCTION

Description:

Input:

SATURATION_FUNCTION `Name`

> **SATURATION_FUNCTION_TYPE VAN_GENUCHTEN**
>
> **SATURATION_FUNCTION_TYPE MUALEM**
>
> **PERMEABILITY_FUNCTION_TYPE VAN_GENUCHTEN**
>
> **PERMEABILITY_FUNCTION_TYPE MUALEM**
>
> **RESIDUAL_SATURATION** `Value`
>
> **LAMBDA** `Value`
>
> **ALPHA** `Value`
>
> **MAX_CAPILLARY_PRESSURE** `Value`
>
> **BETAC** `Value`
>
> **POWER** `Value`

(., /, END)

Explanation:

Example:

```
SATURATION_FUNCTION sf1
SATURATION_FUNCTION_TYPE VAN_GENUCHTEN
RESIDUAL_SATURATION 0.1d0
LAMBDA 2.67d0
ALPHA 2.042d-4
MAX_CAPILLARY_PRESSURE 1d8
BETAC 0.d0
POWER 1.d0
/
```

5.27 Keyword: SOURCE_SINK

SOURCE_SINK `<name>`

REGION <region_name> name of the region the source/sink term is applied to

FLOW_CONDITION <condition_name> name of the flow condition

TRANSPORT_CONDITION <condition_name> name of the transport condition

(., /, END)

Example:

```
SOURCE_SINK Well_2-9_1
FLOW_CONDITION Injection_1
TRANSPORT_CONDITION Source
REGION Well_2-9_1
/
```

5.28 Keyword: STRATIGRAPHY (STRATA)

STRATIGRAPHY (STRATA)

REGION region_name

MATERIAL material_name

INACTIVE

(., /, END)

Description: the keyword **TIME** controls the simulation time.

Input:

5.29 Keyword: TIME

TIME

> **FINAL_TIME** `Value Unit (s, m, h, d, mo, y)`
>
> **INITIAL_TIMESTEP_SIZE** `Value Unit (s, m, h, d, mo, y)`
>
> **MAXIMUM_TIMESTEP_SIZE** `Value Unit (s, m, h, d, mo, y)`
>
> **MAXIMUM_TIMESTEP_SIZE** `Value Unit (s, m, h, d, mo, y)` **AT** `Value Unit (s, m, h, d, mo, y)`
>
> **STEADY_STATE**

(., /, END)

Explanation:

Example:

```
TIME
FINAL_TIME 100. h
INITIAL_TIMESTEP_SIZE 1.d-3 h
MAXIMUM_TIMESTEP_SIZE 1.d0 h
/
```

5.30 Keyword: TIMESTEPPER

Description: the keyword **TIMESTEPPER** controls time stepping.

Input:

TIMESTEPPER [**FLOW, TRAN, TRANSPORT**]

NUM_STEPS_AFTER_TS_CUT [5]

MAX_STEPS [999999]

TS_ACCELERATION [5]

MAX_TS_CUTS [16]

INITIALIZE_TO_STEADY_STATE

RUN_AS_STEADY_STATE

MAX_PRESSURE_CHANGE [5.d4]

MAX_TEMPERATURE_CHANGE [5.d0]

MAX_CONCENTRATION_CHANGE [1.d0]

MAX_SATURATION_CHANGE [0.5d0]

(., /, END)

Explanation:

Example:

5.31 Keyword: TRANSPORT_CONDITION

Description:

Input:

TRANSPORT_CONDITION `Name`

> **TYPE** [**dirichlet**, **dirichlet_zero_gradient**, **equilibrium**, **neumann**, **mole**, **mole_rate**, **zero_gradient**]
>
> **TIME** `Value`
>
> **UNITS** **s, sec, min, hr, d, day, y, yr**
>
> **CONSTRAINT_LIST**
>
> > `time constraint_name`
>
> **(., /, END)**
>
> **CONSTRAINT** `constraint_name`

(., /, END)

Explanation:

Example:

5.32 Keyword: UNIFORM_VELOCITY Optional

Description:

Input:

UNIFORM_VELOCITY `vlx vly vlz` [m/s]

Explanation: Set uniform velocity for transport mode.

Example:

```
UNIFORM_VELOCITY 3.84259d-6 0.d0 0.d0  ! 1.38333 cm/h
```

5.33 Keyword: USE_TOUCH_OPTIONS

Description:

Input:

USE_TOUCH_OPTIONS {use_touch_options = .true.}

Explanation:

Example:

5.34 Keyword: VELOCITY_DATASET

Description: Set time-dependent velocity for transport mode.

Input:

VELOCITY_DATASET

> **UNITS** cm/h
>
> **CYCLIC**
>
> **INTERPOLATION** **step** [default]
>
> **INTERPOLATION** **linear**
>
> **VELOCITY**
>
> > `Time velx vely velz`
>
> **(., /, END)**

(., /, END)

Explanation:

Example:

```
VELOCITY_DATASET
UNITS cm/h
CYCLIC ! cycles the data set using last time as offset
:INTERPOLATION STEP ! interpolation method (step [default] or linear)
VELOCITY
:time velx vely velz
:time units = time unit in velocity units
0.d0       1.38333d0 0.d0 0.d0
12.d0     -1.38333d0 0.d0 0.d0
24.d0      1.38333d0 0.d0 0.d0
/
/
```

5.35 Keyword: WALLCLOCK_STOP

WALLCLOCK_STOP wallclock_stop_time

Explanation:

Example:

6 Example Input Files

6.1 Richards Equation

```
:Description: 1D test problem for tracer transport

:=========================== flow mode ========================================
MODE RICHARDS

:=========================== chemistry ========================================
CHEMISTRY
PRIMARY_SPECIES
Tracer
/
/

:=========================== solver options ===================================
TIMESTEPPER
MAX_STEPS 10
TS_ACCELERATION 8
/

NEWTON_SOLVER FLOW
:RTOL 1.d-4
:ATOL 1.d-4
:STOL 1.e-60
:DTOL 1.e4
:ITOL_UPDATE 1.d0
:NO_INFINITY_NORM
::NO_PRINT_CONVERGENCE
:PRINT_DETAILED_CONVERGENCE
/

LINEAR_SOLVER FLOW
KSP_TYPE GMRES
PC_TYPE NONE
:KSP_TYPE PREONLY
:PC_TYPE LU
:SOLVER GMRES
/

NEWTON_SOLVER TRANSPORT
:RTOL 1.d-4
:ATOL 1.d-4
:STOL 1.e-60
:DTOL 1.e4
:ITOL_UPDATE 1.d-4
NO_INFINITY_NORM
:NO_PRINT_CONVERGENCE
:PRINT_DETAILED_CONVERGENCE
/

LINEAR_SOLVER TRANSPORT
KSP_TYPE GMRES
```

```
PC_TYPE NONE
:KSP_TYPE PREONLY
:PC_TYPE LU
:SOLVER GMRES
/

:=========================== discretization ===================================
GRID
TYPE structured
ORIGIN 0.d0 0.d0 0.d0
NXYZ 32 32 32
BOUNDS
0.d0 100.d0
0.d0 100.d0
0.d0 100.d0
/
END

:=========================== fluid properties =================================
FLUID_PROPERTY
DIFFUSION_COEFFICIENT 1.d-9
/

:=========================== material properties ==============================
MATERIAL_PROPERTY soil1
ID 1
POROSITY 0.25d0
TORTUOSITY 1.d0
SATURATION_FUNCTION default
PERMEABILITY
PERM_X 1.d-12
PERM_Y 1.d-12
PERM_Z 1.d-12
/
/

MATERIAL_PROPERTY soil2
ID 2
POROSITY 0.25d0
TORTUOSITY 1.d0
SATURATION_FUNCTION default
PERMEABILITY
PERM_X 5.d-13
PERM_Y 5.d-13
PERM_Z 5.d-13
/
/

:=========================== saturation functions =============================
SATURATION_FUNCTION default
/

:=========================== output options ===================================
OUTPUT
:MASS_BALANCE
TIMES y 0.25d0 0.5d0 0.75d0
```

```
FORMAT TECPLOT BLOCK
VELOCITIES
/

:=========================== times ===========================================
TIME
FINAL_TIME 1.d0 y
INITIAL_TIMESTEP_SIZE 1.d-3 y
MAXIMUM_TIMESTEP_SIZE 1.d-1 y
/

:=========================== regions =========================================
REGION all
COORDINATES
0.d0 0.d0 0.d0
100.d0 100.d0 100.d0
/
END

REGION top_layer
COORDINATES
0.d0 0.d0 60.d0
100.d0 100.d0 100.d0
/
END

REGION bottom_layer
COORDINATES
0.d0 0.d0 0.d0
100.d0 100.d0 60.d0
/
END

REGION west
FACE WEST
COORDINATES
0.d0 0.d0 0.d0
0.d0 100.d0 100.d0
/
END

REGION east
FACE EAST
COORDINATES
100.d0 0.d0 0.d0
100.d0 100.d0 100.d0
/
END

REGION north
FACE NORTH
COORDINATES
0.d0 100.d0 0.d0
100.d0 100.d0 100.d0
/
END
```

```
REGION south
FACE SOUTH
COORDINATES
0.d0 0.d0 0.d0
100.d0 0.d0 100.d0
/
END

REGION top
FACE TOP
COORDINATES
0.d0 0.d0 100.d0
100.d0 100.d0 100.d0
/
END

REGION bottom
FACE BOTTOM
COORDINATES
0.d0 0.d0 0.d0
100.d0 100.d0 0.d0
/
END

REGION well
COORDINATES
50.d0 50.d0 50.d0
50.d0 50.d0 50.d0
/
END

:=========================== flow conditions ==================================
FLOW_CONDITION initial
TYPE
PRESSURE hydrostatic
/
PRESSURE 1956741.84 ! 200 meter piezometric head (200*997.32*9.81)
/

FLOW_CONDITION source
TYPE
RATE volumetric_rate
/
RATE 2.77777778d-3  ! 10 m^3/hr
/

:=========================== transport conditions =============================
TRANSPORT_CONDITION initial
TYPE zero_gradient
CONSTRAINT_LIST
0.d0 initial
/
END

TRANSPORT_CONDITION source
```

```
TYPE dirichlet
CONSTRAINT_LIST
0.d0 well
/
/

:=========================== constraints =====================================
CONSTRAINT well
CONCENTRATIONS
Tracer 1.d0 T
/
END

CONSTRAINT initial
CONCENTRATIONS
Tracer 1.d-40 T
/
END

:=========================== condition couplers ===============================
: initial condition
INITIAL_CONDITION
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION all
END

: west boundary condition
BOUNDARY_CONDITION west
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION west
END

: east boundary condition
BOUNDARY_CONDITION east
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION east
END

: north boundary condition
BOUNDARY_CONDITION north
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION north
END

: south boundary condition
BOUNDARY_CONDITION south
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION south
END

: top boundary condition
```

```
BOUNDARY_CONDITION top
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION top
END

: bottom boundary condition
BOUNDARY_CONDITION bottom
FLOW_CONDITION initial
TRANSPORT_CONDITION initial
REGION bottom
END

: well source/sink
SOURCE_SINK well
FLOW_CONDITION source
TRANSPORT_CONDITION source
REGION well
END

:=========================== stratigraphy couplers ============================
STRATA
REGION top_layer
MATERIAL soil1
END

STRATA
REGION bottom_layer
MATERIAL soil1
END
```

6.2 Carbon Sequestration Problem with Reaction with Calcite

```
MODE FLASH2
:MODE MPHASE

:CHECKPOINT 100
:RESTART pflotran.chk800

:=========================== discretization ===================================
GRID
TYPE structured
ORIGIN 0.d0 0.d0 0.d0
NXYZ 160  160  25
BOUNDS
0.d0 7000.d0
0.d0 7000.d0
0.d0 250.d0
/
END

:PROC 2 1 1
```

```
:=========================== solver options ==================================
TIMESTEPPER
:MAX_STEPS 50
TS_ACCELERATION 8
MAX_PRESSURE_CHANGE 5.D4
MAX_TEMPERATURE_CHANGE 2.D0
MAX_CONCENTRATION_CHANGE 0.005
MAX_SATURATION_CHANGE  0.01
/

NEWTON_SOLVER FLOW
ATOL 1D-8
RTOL 1D-8
STOL 1D-30
DTOL 1D15
ITOL 1D-8
MAXIT 25
MAXF 100
END
:

:=========================== times ===========================================
TIME
FINAL_TIME 300.d0 y
INITIAL_TIMESTEP_SIZE 1.d-3 y
MAXIMUM_TIMESTEP_SIZE 0.05 y at 10 y
MAXIMUM_TIMESTEP_SIZE 0.1 y at 100 y
MAXIMUM_TIMESTEP_SIZE 0.25 y at 200 y
MAXIMUM_TIMESTEP_SIZE 0.5 y at 300 y
MAXIMUM_TIMESTEP_SIZE 1. y at 500 y
MAXIMUM_TIMESTEP_SIZE 2.5 y at 1000 y
/

:=========================== output options ==================================
OUTPUT
MASS_BALANCE
TIMES y 10. 50. 100. 200. 300.
FORMAT TECPLOT BLOCK
:FORMAT TECPLOT POINT
VELOCITIES
/

:=========================== fluid properties ================================
FLUID_PROPERTY
DIFFUSION_COEFFICIENT 1.d-9
/

:=========================== material properties =============================
MATERIAL_PROPERTY rock2
ID 1
POROSITY 0.38d0
TORTUOSITY 1d-1
ROCK_DENSITY 2.65E3
SPECIFIC_HEAT 1E3
THERMAL_CONDUCTIVITY_DRY 0.5
THERMAL_CONDUCTIVITY_WET 0.5
```

```
SATURATION_FUNCTION sf2
PERMEABILITY
PERM_X 2.d-12
PERM_Y 2.d-12
PERM_Z 2.d-12
/
/

:=========================== saturation functions ============================
SATURATION_FUNCTION sf2
PERMEABILITY_FUNCTION_TYPE PRUESS_1
SATURATION_FUNCTION_TYPE PRUESS_1
RESIDUAL_SATURATION LIQUID_PHASE 0.25
RESIDUAL_SATURATION GAS_PHASE 0.1
LAMBDA 0.75d0
ALPHA 5d-3
MAX_CAPILLARY_PRESSURE 1.d6
BETAC 2.d0
POWER 1.d0
/

:=========================== regions =========================================
REGION all
COORDINATES
0.d0   0.d0 0.d0
7000.D0 7000.D0 250.D0
/
END

REGION west
FACE WEST
COORDINATES
0.d0 0.d0 0.d0
0.d0 7000.d0 250.d0
/
END

REGION east
FACE EAST
COORDINATES
7000.d0 0.d0 0.d0
7000.d0 7000.d0 250.d0
/
END

REGION well
BLOCK 77 84 77 84 5 5
:COORDINATES
:50.d0 0.d0 50.d0
:51.d0 1.d0 51.d0
/
:END

:=========================== flow conditions =================================
FLOW_CONDITION initial
UNITS Pa,C,M,yr
```

```
TYPE
PRESSURE hydrostatic
TEMPERATURE zero_gradient
CONCENTRATION zero_gradient
/
IPHASE 1
PRESSURE 2D7 2D7
TEMPERATURE 50 C
CONCENTRATION 1d-6 M
ENTHALPY 0.d0 0.d0
/

skip
FLOW_CONDITION top
UNITS Pa,C,M,yr
TYPE
PRESSURE dirichlet
TEMPERATURE zero_gradient
CONCENTRATION zero_gradient
/
IPHASE 1
PRESSURE 2D7 2D7
TEMPERATURE 50
CONCENTRATION 1d-6
ENTHALPY 0.d0 0.d0
/
noskip

FLOW_CONDITION side
UNITS Pa,C,M,yr
TYPE
PRESSURE hydrostatic
:PRESSURE dirichlet
TEMPERATURE zero_gradient
CONCENTRATION zero_gradient
/
IPHASE 1
PRESSURE 2D7 2D7
TEMPERATURE 50
CONCENTRATION 1d-6
ENTHALPY 0.d0 0.d0
/

FLOW_CONDITION source
UNITS Pa,C,M,yr
TYPE
PRESSURE mass_rate
TEMPERATURE dirichlet
CONCENTRATION dirichlet
/
PRESSURE file c2i.txt  ! kg/s
TEMPERATURE 50.d0
CONCENTRATION 0.D0
ENTHALPY 0.d0 0.d0
/
```

```
:=========================== condition couplers ===============================
: initial condition
INITIAL_CONDITION
FLOW_CONDITION initial
REGION all
END

: top boundary condition
:BOUNDARY_CONDITION top
:FLOW_CONDITION top
:REGION top
:END

: boundary condition
BOUNDARY_CONDITION east
FLOW_CONDITION side
REGION east
END

BOUNDARY_CONDITION west
FLOW_CONDITION side
REGION west
END

SOURCE_SINK
FLOW_CONDITION source
REGION well
END

:=========================== stratigraphy couplers ============================
STRATA
REGION all
MATERIAL rock2
END
```

7 Adaptive Mesh Refinement: SAMRAI

7.1 Creating the SAMRAI input file

```
DIRICHLET = 0
NEUMANN   = 1
PERIODIC  = 2
MIXED     = 3
ROBIN     = 4

// name of pflotran specific input file
pflotran_filename="pflotran.15c.in"

CartesianGeometry{
  //  Specify lower/upper corners of the computational domain and a
  //  set of non-overlapping boxes defining domain interior.  If union
```

```
  //  of boxes is not a parallelpiped, lower/upper corner data corresponds
  //  to min/max corner indices over all boxes given.
  //  x_lo  -- (double array) lower corner of computational domain [REQD]
  //  x_up  -- (double array) upper corner of computational domain [REQD]
  //  domain_boxes  -- (box array) set of boxes that define interior of
  //                   physical domain. [REQD]
  //  periodic_dimension -- (int array) coordinate directions in which
  //                        domain is periodic.  Zero indicates not
  //                        periodic, non-zero value indicates periodicity.
  //                        [0]
  // indexing always starts at 0 for domain_boxes => 0-79 = 80 cells
  domain_boxes = [(0,0,0), (19,19,79)]
   x_lo         = 0.e0, 0.e0, 0.e0
   x_up         = 20.e0, 20.e0, 80.e0
}

GriddingAlgorithm{
   // Information used to create patches in AMR hierarchy.
   // max_levels -- (int) max number of mesh levels in hierarchy [REQD]
   //
   // For most of the following parameters, the number of prescribed data
   // values need not match the number of levels in the hierarchy
   // (determined by max_levels).  If more values are given than number
   // of levels, extraneous values will be ignored.  If less are given, then
   // values that correspond to individual levels will apply to those
   // levels.  Missing values will be taken from those for the finest
   // level specified.
   //
   // ratio_to_coarser {
   //   level_1 -- (int array) ratio between index spaces on
   //              level 1 to level 0 [REQD]
   //   level_2 -- (int array)  ratio between index spaces on
   //              level 2 to level 1 [REQD]
   //   etc....
   // }
   // largest_patch_size {
   //   level_0 -- (int array) largest patch allowed on level 0.
   //              [REQD]
   //   level_1 -- (int array)    "       "      "   "  level 1
   //              [level 0 entry]
   //   etc....
   // }
   max_levels = 7
   largest_patch_size {
      level_0 = 20, 20, 80
      level_1 = 40, 40, 40
      // all finer levels will use same values as level_1...
   }
   smallest_patch_size {
      level_0 = 10, 10, 40
      level_1 =  5,  5, 10
      // all finer levels will use same values as level_1...
   }
   ratio_to_coarser {
      level_1  = 2, 2, 2
      // all finer levels will use same values as level_1...
```

```
   }
   combine_efficiency = 0.95
   efficiency_tolerance = 0.95
   coalesce_boxes=FALSE
   check_nonrefined_tags='i'
}

StandardTagAndInitialize{
   tagging_method = "REFINE_BOXES"
   RefineBoxes{
      // index region on level 0 to tag for refinement in level 0 indices.
      // the full index region on level 0 is [(0,0,0),(19,19,79)]
      // Note that this will create a refinement region on level 1 extending from
      // [(0,0,60*2), (19*2+1,19*2+1,79*2+1)] when refined by a factor of 2 in
      // each direction. The full index space on level one is [(0,0,0), (2*19+1,2*19+1,2*79+1)]
      level_0 = [(0,0,60),(19,19,79)]          // 0.5 m
      level_1 = [(0,0,140),(39,39,159)]        // 0.25
      level_2 = [(0,0,300),(79,79,319)]        // 0.125
      level_3 = [(0,0,620),(159,159,639)]      // 0.0625
      level_4 = [(0,0,1260),(319,319,1279)]    // 0.03125
      level_5 = [(0,0,2540),(639,639,2559)]    // 0.015625
   }
}

LoadBalancer{
   bin_pack = "SPATIAL"
}

PflotranApplicationStrategy{
   DriverMode                       = 1
   nl_tangential_coarse_fine_scheme = "LINEAR"
   nl_normal_coarse_fine_scheme     = "LINEAR"
   number_solution_components       = 3

   PflotranMultilevelOperator{
       operator_name                   = "PflotranMultilevelOperator"
       tangent_interp_scheme           = "LINEAR"
       normal_interp_scheme            = "LINEAR"
       adjust_cf_coefficients          = FALSE
       interpolate_ghost_values        = TRUE
       extrapolation_order             = 2
       boundary_conditions             = NEUMANN, NEUMANN, NEUMANN, NEUMANN, NEUMANN, DIRICHLET
       coarsen_diffusive_fluxes        = TRUE
       coarsen_convective_fluxes       = TRUE
       print_info_level                = 0
       cell_refine_op                  = "CONSTANT_REFINE"
       cell_coarsen_op                 = "CONSERVATIVE_COARSEN"
       cell_soln_coarsen_op            = "CONSERVATIVE_COARSEN"
       cell_src_coarsen_op             = "SUM_COARSEN"
       face_refine_op                  = "CONSTANT_REFINE"
       face_coarsen_op                 = "CONSERVATIVE_COARSEN"
       variable_order_interpolation    = FALSE
       use_cf_interpolant              = TRUE
   }

   PflotranFlowPreconditioner{
```

```
    preconditioner_print_flag = FALSE

    pc_solver{
      solver_name                    = "FAC"
      max_iterations                 = 2
      max_error                      = 1.e-12
      print_info_level               = 1
      presolve_iterations            = 2
      postsolve_iterations           = 2
      use_visit                      = FALSE
      viz_directory                  = "fac_viz"
      zero_initial_guess             = TRUE

      level_solver_0 {
           solver_name               = "CellHYPRELevelSolver"
           print_info_level          = 0
           mg_solver_type            = 1
           max_iterations            = 1
           max_error                 = 1.e-12
           num_presweeps             = 2
           num_postsweeps            = 2
           preprocess_rhs            = FALSE
      }

      level_solver_1 {
           solver_name               = "CellLevelSmoother"
           smoother_type             = "Red-black Gauss-Seidel"
           print_info_level          = 0
           max_iterations            = 1
           max_error                 = 1.e-12
           num_sweeps                = 10
      }
    }
  }

  PflotranTransportPreconditioner{

    preconditioner_print_flag = FALSE

    pc_solver{
      solver_name                    = "FAC"
      max_iterations                 = 2
      max_error                      = 1.e-12
      print_info_level               = 1
      presolve_iterations            = 2
      postsolve_iterations           = 2
      use_visit                      = FALSE
      viz_directory                  = "fac_viz"
      zero_initial_guess             = TRUE

      level_solver_0 {
           solver_name              = "CellHYPRELevelSolver"
           print_info_level         = 0
           mg_solver_type           = 1
           max_iterations           = 1
```

```
            max_error                  = 1.e-12
            num_presweeps              = 2
            num_postsweeps             = 2
            preprocess_rhs             = FALSE
       }

       level_solver_1 {
            solver_name                = "CellLevelSmoother"
            smoother_type              = "Red-black Gauss-Seidel"
            print_info_level           = 0
            max_iterations             = 1
            max_error                  = 1.e-12
            num_sweeps                 = 10
       }
     }
   }
}


TimerManager{
   timer_list = "xfer::*::*", "FAC::*::*", "AMRUtilities::*::*", "apps::main::main", \
   "RD2T::*::*", "LevelSolver::*::*"
}
```

8 References

Balay S, Eijkhout V, Gropp WD, McInnes LC and Smith BF (1997) Modern Software Tools in Scientific Computing, Eds. Arge E, Bruaset AM and Langtangen HP (Birkhaüser Press), pp. 163–202.

Goode, D.J. (1996) Direct simulation of groundwater age, Water Resources Research, 32, 289–296.

Hammond, G.E., P.C. Lichtner, C. Lu, and R.T. Mills (2011) PFLOTRAN: Reactive Flow & Transport Code for Use on Laptops to Leadership-Class Supercomputers, Editors: Zhang, F., G. T. Yeh, and J. C. Parker, *Ground Water Reactive Transport Models*, Bentham Science Publishers. ISBN 978-1-60805-029-1.