PhreeqcRM: A reaction module for transport simulators based on the geochemical model PHREEQC

David L. Parkhurst\*, U.S. Geological Survey, Box 25046, MS 413, Denver Federal Center, Denver, CO, 80225, USA, [dlpark@usgs.gov](mailto:dlpark@usgs.gov)

and

Laurin Wissmeier, AF‑Consult Switzerland Ltd, Groundwater Protection and Waste Disposal, Täfernstrasse 26, CH‑5405 Baden, Schweiz, [laurin.wissmeier@afconsult.com](mailto:laurin.wissmeier@afconsult.com)

# Abstract

PhreeqcRM is a geochemical reaction module designed specifically to perform equilibrium and kinetic reaction calculations for reactive transport simulators that use an operator‑splitting approach, that is, simulators that separate the transport and reaction calculations. The basic function of the reaction module is to take component concentrations for the model cells of the transport simulator, run geochemical reactions, and return revised component concentrations to the transport simulator. If multicomponent diffusion is modeled (e.g., Nernst‑Planck equation), then aqueous species concentrations can be used instead of component concentrations. The reaction capabilities are a complete implementation of the reaction capabilities of PHREEQC. The reaction module maintains the composition of all of the reactants that react to equilibrium in each cell, which may include minerals, exchangers, surface complexers, a gas phase, and solid solutions, and reactants that react kinetically, which may include mineral dissolution and precipitation, biological transformations, or other geochemical kinetic process. PhreeqcRM assigns initial and boundary conditions for model cells based on standard PHREEQC input definitions (files or strings) of chemical compositions of solutions and reactants. Additional PhreeqcRM capabilities include methods to account for inactive grid cells, obtain selected results, print detailed reaction calculations for selected cells, and save the chemical state of a calculation. The module demonstrates good scalability for parallel processing by using either multithreading with OpenMP on shared memory systems, or multiprocessing with MPI (Message Passing Interface) on distributed memory systems. PhreeqcRM is written in C++, but interfaces allow methods to be called from C or Fortran90. By using the PhreeqcRM reaction module, an existing multicomponent transport simulator can be extended to simulate a wide range of geochemical reactions.

Keywords:

Reactive transport modeling, PHREEQC, PHAST, FEFLOW, reaction engine, parallelization, contaminant transport, , MoMaS reactive transport benchmark

# Introduction

During the past decade, one of the main driving forces in reactive transport modelling has been the ongoing global search for strategies of safe nuclear waste disposal [1–9]. Governmental agencies are taking great efforts to manage radioactive waste and to assess the risks of different disposal schemes. Reactive transport modelling is of interest because of (1) its potential to predict the evolution of contaminant plumes over long time scales, and (2) its capabilities to simulate near-field processes, which are used to improve safety in the design of confining structures and containers. Yet, the quest for safe radioactive waste disposal is only one of many environmental issues where reactive transport modelling has become an essential decision support and planning tool.

With the search for alternative forms of energy production, reactive transport modelling is increasingly used to predict long‑term effects of geothermal energy usage [10–13]. The mining industry and its regulatory authorities routinely use reactive transport modelling to optimize exploitation strategies, assess the risk for groundwater contamination, and design rehabilitation schemes [14–18]. Particularly with respect to emerging technologies, such as hydraulic fracturing and shale‑gas extraction, reactive transport modelling is a useful tool to estimate profitability and evaluate adverse effects on the environment. Furthermore, reactive transport modelling has been employed successfully to demonstrate the feasibility, dangers, and uncertainties of underground carbon dioxide storage [19–24]. All of these fields need an accurate description of flow and transport in combination with reaction processes, and couplings of existing transport simulators with established geochemistry packages have become increasingly popular [25,26] as the means to provide these simulation capabilities.

With the release of IPhreeqc [27]—a general purpose application programming interface (API) for the geochemical modelling framework PHREEQC [28]—the necessary functionality to use PHREEQC as a reaction engine for transport simulators became available. Since its release, IPhreeqc has been coupled to numerous transport codes in a wide variety of contexts. Wissmeier and Barry [29] were the first to develop a coupling with COMSOL, which gives access to the full range of COMSOL’s environmental flow and transport capabilities together with the complete set of geochemical reactions in PHREEQC. In addition, they outlined a general strategy for couplings with different flow and transport simulators. On these grounds, Nardi et al. [30] have published another COMSOL‑IPhreeqc coupling. The free environmental flow‑ and transport‑modelling platform OpenGeoSys [31] has an interface to IPhreeqc for coupled thermo‑hydro‑mechanical‑chemical (THMC) simulations. Specialized tools using IPhreeqc have been developed by Takahashi and Ishida [32] for cementitious materials and Huber et al. [33] for the paper-making process. More recently, Patel et al. [34] implemented IPhreeqc as reaction engine for pore‑scale multicomponent reactive transport that used a Lattice‑Boltzmann approach. This non‑comprehensive list indicates scientific and industrial interest in using PHREEQC as reaction module.

However, IPhreeqc has limitations for couplings with transport simulators. Its API provides access to all of PHREEQC’s reaction capabilities by reading and interpreting PHREEQC script commands from input strings or files. Although this approach provides a comprehensive interface, it requires substantial coding in the client software to compose PHREEQC scripts for exchanging data and running reactions. In addition, the extra computer time necessary to process the scripts will degrade performance. In this paper, we present a new reaction module based on IPhreeqc called PhreeqcRM that is specifically designed for couplings with environmental flow and transport simulators. It provides a high‑level interface that allows multicomponent transport codes to implement geochemical reactions with a minimum amount of programming, while maintaining the full functionality of PHREEQC reaction capabilities.

The paper has two main parts: In section 2, we provide a generic program flow for the coupling of PhreeqcRM with flow and transport simulators together with detailed information on the key methods of its API. In section 3, we present the implementation of PhreeqcRM as reaction engine for the groundwater simulators PHAST [35] and FEFLOW [36]. The codes are verified through two test cases: a 3D (three‑dimensional) analytical solution by Sun et al. [37] and the MoMaS (Modeling, Mathematics and numerical Simulations related to nuclear waste management problems) reactive transport benchmark of GdR (Groupement de Recherche) [38].

# The PhreeqcRM API

PHREEQC is a general‑purpose geochemical reaction model that has capabilities to model interactions between water and one or more reactants, including minerals, gases, ion exchangers, surface complexers, solid solutions, and kinetic reactants. PHREEQC relies on a set of keyword data blocks to define the types and compositions of the reactants. IPhreeqc [27] encapsulates the capabilities of PHREEQC in a C++ class (with wrappers for C and Fortran) to facilitate integration of PHREEQC into other computer programs. However, IPhreeqc relies primarily on processing strings that define keyword data blocks and arrays of selected-output data to automate the use of PHREEQC. For reactive transport modelling, translating cell solutions to strings and processing output arrays item‑by‑item are tedious programming efforts and inefficient for running millions of calculations.

PhreeqcRM is a C++ class that encapsulates IPhreeqc and is designed specifically for performing reaction calculations in reactive transport simulators. The methods of the class allow data arrays to be transferred and reaction calculations to be performed without the use of keyword data blocks. Selected‑output data can be obtained as an array for visualization and output to files. The methods in PhreeqcRM include all of the capabilities necessary to implement PHREEQC reaction calculations in a multicomponent transport simulator.

## Implementation of PhreeqcRM in a Multicomponent Transport Simulator

PhreeqcRM is intended to be the reaction-calculation engine in an operator‑splitting approach to reactive transport. The simplest operator‑splitting approach is the sequential non‑iterative approach (SNIA) [39–41], which separates the simultaneous processes of solute transport and reaction into alternating transport and reaction steps with predefined time steps. PhreeqcRM is not restricted to SNIA, but SNIA is assumed here for simplicity.

PHREEQC uses moles of elements, including oxygen and hydrogen, as the definition of a solution. To account for charge imbalance in boundary and initial solutions and to compensate for charge imbalances that may be induced by surface‑complexation reactions, charge imbalance is also required for a complete solution definition. The elements plus charge imbalance are referred to as components. PhreeqcRM does not rely on a specific form of the transport equations. If the same form of transport equation applies for all dissolved species, and equation parameters are species‑independent, then the aqueous solution may be transported in terms of components rather than solution species, which reduces the number of transported entities.

Alternatively, PhreeqcRM provides functionality to retrieve the solution composition in terms of aqueous species, together with species charge and diffusion coefficient, to facilitate couplings that account for species‑dependent transport processes (e.g., Nernst‑Planck equation). In this case, it is necessary for the transport equations to account for each aqueous species, while maintaining a net current of zero [7]. In the following sections, we consider only component‑based transport with species‑independent transport parameters.

Numerical solutions of solute transport may lead to negative concentrations, typically close to sharp concentration fronts. Because PHREEQC calculates reactions in terms of moles, negative concentrations from the transport step have to be amended (set to zero) during the reaction step. To avoid significant mass‑balance errors, negative concentrations from the previous transport step can be added to the result of the reaction step prior to updating concentrations for the following transport step. This approach should conserve mass; however, errors may accumulate if reactions are not calculated for a component because concentrations are set to zero.

A generic program flow for the implementation of PhreeqcRM as a reaction engine for a transport simulator is displayed in Figure 1. The main Simulation process accounts for the storage of coupling settings and visualization of reaction results in the host application. The specific tasks for PhreeqcRM that the transport simulator needs to execute during the *Initialization,* *Reactions*, and *Finalize* processes are detailed in the following section.



Figure 1: Generic program flow for a reactive transport simulator using operator splitting.

## Key Methods of the PhreeqcRM API

The use of PhreeqcRM entails a sequence of method calls that create and initialize the instance, set initial conditions and reactions for the model cells, determine concentrations for boundary conditions, run reaction calculations for a series of time steps, and finalize the instance. Although C++ methods are described, equivalent C and Fortran subroutines, generally with the same names preceded by “RM\_”, are available for transport codes written in these languages. (PhreeqcRM is available at <http://wwwbrr.cr.usgs.gov/projects/GWC_coupled/phreeqc>. Included in the distributions is documentation of all methods of the API and simple advection examples in C, Fortran, and C++. Tables 1-7 are excerpts from the C++ example, and table 8 is from the Fortran example.)

### Create and Initialize PhreeqcRM

As a C++ class, a constructor creates an instance of PhreeqcRM. Assuming that the source code has been compiled for use with multiple threads (with USE\_OPENMP as pre‑processor definition), the number of threads (*nthreads*) and the number of transport cells (*nxyz*) that represent the nodes or cells in the client transport simulator are parameters for the constructor. For multiple processes (with USE\_MPI as pre‑processor definition), the constructor requires an MPI (Message Passing Interface) communicator and the number of transport cells. In Fortran or C, the subroutine RM\_Create is used to create the instance.

The action taken when errors occur can be set to one of three options: (1) the invoked method returns an error code, (2) the program throws an exception, or (3) the program aborts (SetErrorHandlerMode method). A prefix may be defined for the names of the output (*prefix.chem.txt*) and log files (*prefix.log.txt*) by using the SetFilePrefix method, and these files are opened with the OpenFiles method.

The transport simulator must transport each component and provide new concentrations to PhreeqcRM at each time step. The minimum set of chemical components for PhreeqcRM is H (hydrogen), O (oxygen), and charge (imbalance). However, because the number of moles of water is large compared to all other solutes, the non‑water H and O concentrations are several orders of magnitude smaller than the total concentrations, yet PHREEQC requires accurate accounting of the non‑water H and O. It is numerically advantageous to consider water as a separate component; therefore, the default set of components is water, total H minus H in water, total O minus O in water, and charge. Water can be included or excluded as a separate component by the use of the method SetComponentWater. Often the calculation time for transport calculation is small relative to the reaction calculations, so the expense of one extra component transport for water (the default) is minimal.

The number of cells in the reaction module may differ from the number of cells in the transport simulator because of chemically inactive zones or symmetry. A many‑to‑one mapping can be defined that translates each transport‑cell number to a reaction‑cell number, such that the number of reaction cells is less than or equal to the number of transport cells (CreateMapping method). Regardless of the mapping, PhreeqcRM returns results for every transport cell.

The reaction cells are defined to have a representative volume, , of 1 L by default. A solution defined in PhreeqcRM input is adjusted (maintaining the same concentrations) to a volume equal to the water content, (L), according to

|  |  |
| --- | --- |
|  |  |

where is the liquid phase saturation (unitless), is porosity (unitless). The saturation, porosity, and representative volume are set with the methods SetSaturation, SetPorosity, and SetRepresentativeVolume, respectively.

The concentration units of dissolved constituents, , used in the transport simulator can be (mg/L), (mol/L), or (kg/kg solution); the method SetUnitsSolution specifies the concentration units of the transport simulator. The concentrations from the transport simulator are converted in PhreeqcRM to moles per cell () by using the following equations:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

where is gram formula weight of component/species (g/mol) and is the solution density (kg/L) as set by the SetDensity method.

To convert from solid‑phase moles in the PhreeqcRM input, , to the moles of solid phases in the representative volume of the reaction cell, (mol), one of the following definitions can be selected:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

With Eq. (6), the solid‑phase input in PhreeqcRM () is mol/L of porous media, with Eq. (7), is mol/L of pore volume (fully saturated water volume); and with Eq. (8), is mol/L of solid volume. The conversion from solid‑ and gas‑phase input in a PHREEQC input file to moles in the cell using one of these three definitions is specified by the methods SetUnitsPPassemblage, SetUnitsExchange, SetUnitsSurface, SetUnitsGasPhase, SetUnitsSSassemblage, and SetUnitsKinetics.

Kinetic reactions in PHREEQC are defined as the transfer of components to or from the solution in terms of moles. Care must be taken to convert rate expressions (PHREEQC RATES data block) from change in concentration per time to molar change per time by multiplying by the current solution volume (PHREEQC Basic function SOLN\_VOL).

A sample of C++ code that illustrates the creation of a PhreeqcRM instance and its initialization is displayed in Table 1.

Table 1: Create and initialize PhreeqcRM.

|  |
| --- |
| int nxyz = 40;  int nthreads = 3;  PhreeqcRM phreeqc\_rm(nxyz, nthreads);  phreeqc\_rm.SetErrorHandlerMode(1);  phreeqc\_rm.SetComponentH2O(false);  phreeqc\_rm.SetFilePrefix("Advect\_cpp");  phreeqc\_rm.OpenFiles();  // grid2chem is a mapping from transport cells to reaction cells  std::vector<int> grid2chem;  grid2chem.resize(nxyz, ‑1);  for (int i = 0; i < nxyz / 2; i++)  {  grid2chem[i] = i;  grid2chem[i + nxyz / 2] = i;  }  status = phreeqc\_rm.CreateMapping(grid2chem);  std::vector<double> sat, por, rv;  sat.resize(nxyz, 1.0);  phreeqc\_rm.SetSaturation(sat);  por.resize(nxyz, 0.2);  phreeqc\_rm.SetPorosity(por);  rv.resize(nxyz, 1.0);  phreeqc\_rm.SetPoreVolume(rv);  phreeqc\_rm.SetUnitsSolution(2);  phreeqc\_rm.SetUnitsPPassemblage(1);  phreeqc\_rm.SetUnitsExchange(1);  phreeqc\_rm.SetUnitsSurface(1);  phreeqc\_rm.SetUnitsGasPhase(1);  phreeqc\_rm.SetUnitsSSassemblage(1);  phreeqc\_rm.SetUnitsKinetics(1); |

### Set Initial Conditions

Initial conditions for solution compositions and solid and gas phase reactants can be read from one or more PHREEQC input files or strings and then distributed to the model cells. An initial solution composition is assigned to each cell. In addition, sets of reactants can be assigned to each cell to define the types of reactions that can occur in the cell (equilibrium phases, cation exchange, surface complexation, solid solutions, gas phase, and kinetics) and the initial number of moles of each reactant, for example the initial amount of a mineral or the number of ion‑exchange sites.

A reaction module has at least three IPhreeqc instances, one or more workers, an input‑processing instance called the InitialPhreeqc instance, and a utility instance, which is optionally used for special calculations. If using MPI, each process has a reaction module with one worker and a total of three IPhreeqc instances. If using OpenMP, the reaction module has a number of workers equal to the number of threads specified in the constructor.

The LoadDatabase method loads a PHREEQC database for all of the IPhreeqc instances. The PHREEQC database defines a set of elements and corresponding thermodynamic data for the aqueous species, gas components, and mineral phases derived from these elements. The database determines which type of aqueous model will be used—the WATEQ ion‑association, LLNL (Lawrence Livermore National Laboratory) ion‑association, Pitzer, or SIT (Specific Interaction Theory) model. The database may also contain thermodynamic data for ion‑exchange and surface‑complexation processes and rate expressions for kinetic reactions. Like PHREEQC, it is possible to modify or augment the definitions of the database with other input files; however, the thermodynamic definitions must be consistent for all of the IPhreeqc instances.

The RunFile method reads and executes a PHREEQC input file by any combination of IPhreeqc instances: worker, InitialPhreeqc, and utility. If only initial or boundary conditions are included in the input file, then only the InitialPhreeqc instance needs to run the file. If additions to the database are part of an input file, then all of the IPhreeqc instances should run the file so that all instances use the same thermodynamic data and aqueous model. Selected‑output definitions (combinations of the PHREEQC data blocks SELECTED\_OUTPUT and USER\_PUNCH) need to be read by the worker instances to allow extraction of selected data for visualization or output to files.

The strategy for assigning initial conditions to the reaction cells is to transfer definitions from the InitialPhreeqc instance to the worker instances. Once the InitialPhreeqc instance reads a PHREEQC input file, a set of solutions and reactants is present within the InitialPhreeqc instance, each identified by a user number (integer), as specified in the input file. These solutions and reactants can be distributed to the reaction cells by use of the InitialPhreeqc2Module method. The argument to the method is an array of integers that specifies the solutions and reactants, as identified by user numbers, that are assigned to each transport cell. The array is indexed as follows: (1) solution, (2) equilibrium phases, (3) exchangers, (4) surfaces, (5) gas phase, (6) solid solutions, and (7) kinetic reactions. Solution user numbers for each transport cell are stored first in the array, followed by equilibrium phases user numbers, and so on, equivalent to Fortran storage (*nxyz*, 7). A negative integer indicates that a reactant is not present in a cell. The mapping from transport cells to reaction cells is used to move the solutions and reactants to the worker instances and to number them in the reaction‑cell numbering system.

An alternative form of the InitialPhreeqc2Module method defines two user numbers and a mixing fraction for the definitions of each cell. Thus, the solution in a transport cell could be defined as a mixture of two solutions from the InitialPhreeqc instance. Similarly, the set of equilibrium phases in a cell could be defined as a mixture of two sets of equilibrium phases, with fraction *f* times the number of moles in set 1 and fraction *1*‑*f* times the number of moles in set 2. All other types of reactants can be mixed in a similar fashion.

Instead of defining all reactants in all cells with InitialPhreeqc2Module, an alternative way to distribute initial conditions is to run a PHREEQC input file with the InitialPhreeqc instance and select one user number to assign to each cell in a list of transport cells. All of the solutions and reactants with that user number will be distributed to each transport cell in the list.

If definitions of initial conditions (or boundary conditions) are loaded from separate input files, reactants from the previous condition should be removed from the InitialPhreeqc by using RunString(“DELETE; ‑all;”). It is an error if a reaction cell has not been initialized with at least a solution definition.

It is convenient to get a list of components that have been defined in the solutions and reactants of the InitialPhreeqc instance. This list is the set of components that needs to be transported by the transport simulator. The FindComponents method accumulates a list of components that have been used in the InitialPhreeqc instance. The method can be called multiple times, once after each call to RunFile for the InitialPhreeqc instance. The method GetComponents returns a list of all components that have been accumulated, including water, hydrogen, oxygen, charge, and any other element that has been defined in a solution, equilibrium phase assemblage, surface, gas phase, solid solution, or kinetic reaction definition.

Once the initial conditions for solutions and reactions have been defined, it is useful to ensure that the equilibrium reactants (all except kinetic reactants) are equilibrated with the aqueous solution by using the RunCells method. By setting the time step for the RunCells calculation to 0 seconds (SetDeltaTime), no kinetic reactions are run, but all other reactants equilibrate with the solution, perhaps resulting in revised initial conditions for the solutions and the equilibrium reactants. The revised solution compositions can be retrieved with the GetConcentrations method for use in the first transport step. Table 2 displays sample code for the initialization of reaction cells and initial solution conditions.

Table 2: Initial conditions and reaction for each cell.

|  |
| --- |
| phreeqc\_rm.LoadDatabase("phreeqc.dat");  bool workers = true  bool initial\_phreeqc = true  bool utility = true  phreeqc\_rm.RunFile(workers, initial\_phreeqc, utility, "advect.pqi");  std::vector<int> ic1, ic2;  ic1.resize(nxyz \* 7, ‑1);  ic2.resize(nxyz \* 7, ‑1);  std::vector<double> f1;  f1.resize(nxyz \* 7, 1.0);  for (int i = 0; i < nxyz; i++)  {  ic1[i] = 1; // Solution 1  ic1[2 \* nxyz + i] = 1; // Exchange 1  }  status = phreeqc\_rm.InitialPhreeqc2Module(ic1, ic2, f1);  int ncomps = phreeqc\_rm.FindComponents();  const std::vector<std::string> &components = phreeqc\_rm.GetComponents();  std::vector<double> c;  c.resize(nxyz \* components.size());  phreeqc\_rm.SetTime(0.0);  phreeqc\_rm.SetTimeStep(0.0);  phreeqc\_rm.RunCells();  phreeqc\_rm.GetConcentrations(c); |

### Set Boundary Conditions

Boundary‑condition aqueous concentrations also are defined through PHREEQC input strings or files. Component concentrations can be extracted from the InitialPhreeqc instance for use as boundary conditions in transport calculations. The method InitialPhreeqc2Concentrations takes a list of user numbers corresponding to solution definitions in the InitialPhreeqc instance and generates an array of concentrations for each component for each solution. These concentrations may then be applied to the set of boundary conditions, cells or nodes, in the transport simulator. As in the definition of initial conditions, it also is possible to define boundary condition concentrations as a mixture of two solutions. Sample code for the calculation of component boundary concentrations is displayed in Table 3.

Table 3: Boundary conditions.

|  |
| --- |
| std::vector<double> bc\_conc, bc\_f1;  std::vector<int> bc1, bc2;  int nbound = 1;  // bc1 is solution 0, bc2 not used  bc1.resize(nbound, 0);  bc2.resize(nbound, ‑1);  bc\_f1.resize(nbound, 1.0);  phreeqc\_rm.InitialPhreeqc2Concentrations(bc\_conc, bc1, bc2, bc\_f1); |

### Transfer Data to PhreeqcRM

SetTemperature, SetPressure, SetSaturation, SetPorosity, and SetDensity can be used to define the corresponding properties in the reaction cells after a transport step. SetConcentrations transfers the transported concentrations to the reaction module, but uses the saturation, porosity, representative volume, and possibly density (for mass-fraction concentrations) to set the number of moles of elements in each of the reaction cells.

Selected output, as defined by SELECTED\_OUTPUT/USER\_PUNCH data blocks in PHREEQC input, is the primary means for obtaining geochemical output for a time step. Selected output can be activated or inactivated with the SetSelectedOutputOn method. The SetPrintChemistryOn method is intended primarily for debugging. A complete description of the solution and each reactant for each cell is printed to the output file, which results in a very large file if used with a full‑scale simulation. The set of cells that are printed can be limited by use of a Boolean array with the SetPrintChemistryMask method. Sample code for the transfer of data from the transport simulator to PhreeqcRM is displayed in Table 4.

Table 4: Transferring data to PhreeqcRM.

|  |
| --- |
| // spatial arrays  phreeqc\_rm.SetPorosity(por);  phreeqc\_rm.SetSaturation(sat);  phreeqc\_rm.SetTemperature(temperature);  phreeqc\_rm.SetPressure(pressure);  phreeqc\_rm.SetConcentrations(c);  // print parameters  phreeqc\_rm.SetSelectedOutputOn(print\_selected\_output\_on);  phreeqc\_rm.SetPrintChemistryOn(print\_chemistry\_on, false, false); |

### Run Reactions

The current simulation time is set with the method SetTime and the time step for the reaction calculations is set with the method SetDeltaTime. A factor to convert from seconds to user time units in the input to the SetTime and SetDeltaTime methods can be set by the SetTimeConversion method. Equilibrium and kinetic reactions for the time step are run with the RunCells method. Sample code for running a typical reaction step in PhreeqcRM is provided in Table 5.

Table 5: Time stepping and reaction calculations.

|  |
| --- |
| phreeqc\_rm.SetTimeStep(time\_step);  time += time\_step;  status = phreeqc\_rm.SetTime(time);  status = phreeqc\_rm.RunCells(); |

### Transfer Data to the Transport Model

After the reaction calculations, the newly calculated solution volume, solution density, and saturation may be retrieved with GetSolutionVolume, GetDensity, and GetSaturation methods. When the volume of solution increases as result of reactions, the saturation (solution volume divided by the pore volume) may be larger than one. Similarly, if the solution volume decreases, the saturation may be less than one, even for a saturated flow system. The client code can either ignore changes in saturations or transfer the associated solution volume as a source or sink term during the following flow and transport step.

Selected output results can be retrieved with the GetSelectedOutput method (provided it has been enabled with SetSelectedOutputOn). The retrieved selected‑output array can be visualized or written to file. Sample code to transfer data to the transport model after a reaction step is provided in Table 6.

Table 6: Transfer data to transport model.

|  |
| --- |
| std::vector<double> density;  phreeqc\_rm.GetDensity(density);  const std::vector<double> &volume = phreeqc\_rm.GetSolutionVolume();  std::vector<double> so;  phreeqc\_rm.GetSelectedOutput(so); |

### Finalize PhreeqcRM

After time‑stepping and reaction calculations are completed, the output and log files can be closed with the CloseFiles method (Table 7). In C++, the module is destroyed and memory is freed when the destructor is invoked. In Fortran and C, the RM\_Destroy subroutine will destroy the module.

Table 7: Finalize PhreeqcRM

|  |
| --- |
| phreeqc\_rm.CloseFiles(); |

## Parallelization

Operator‑splitting in reactive transport calculations is ideally suited to parallelization because the reaction calculation for each cell is independent of all other cells. Thus, the reaction calculations for the cells can be distributed to multiple processors by using multiple threads, on a shared memory system, or to multiple processes, on a distributed memory system. By using preprocessor definitions (either USE\_OPENMP or USE\_MPI), PhreeqcRM can be compiled to use OpenMP, which enables multiple threads, or MPI, which enables multiple processes. For OpenMP the number of threads is defined when creating a PhreeqcRM instance. For MPI, the number of processes and the set of host computers are defined through arguments to *mpiexec*, thecommand that launches an MPI job. PhreeqcRM has been successfully compiled with OpenMP implementations for Windows® (Visual Studio® 2010/2012/2013) and Scientific Linux and with MPI implementations for Windows® (MSMPI from Microsoft®) and Linux (OPENMPI, version 1.5.4).

With the implementation of multiple threads, each thread has access to all of the data of the reaction module. Several loops within PhreeqcRM are parallelized by using OpenMP directives, most notably a loop in RunCells, where each thread runs reaction calculations on an assigned set of cells. The number of parallelized loops is relatively few but is sufficient to produce good scalability for parallel processing of the reaction calculations on a multiprocessor computer.

The parallelization for multiple processes is complicated by the fact that the data necessary to perform a reaction calculation for a cell must be distributed to the process that is to run reaction calculations for that cell. Particularly, the solution concentrations from the transport simulation must be distributed to the processes at each time step so that the new concentrations are used in the reaction calculations. PhreeqcRM uses a client‑server model to perform a variety of distributed tasks, including reaction calculations. The root process is the client (manager), and the non‑client processes are the servers (workers). For example, when the method RunCells is called, the manager sends a message to the workers that a RunCells task must be done. The workers interpret the message in MpiWorker and run the RunCells method. Within the RunCells method, the manager and workers do the reaction calculations for the set of cells for which each is responsible. The workers then wait for the next task message in MpiWorker. This same manager‑worker structure is used by each method that requires action by the workers, including methods that distribute data from the manager to the workers, such as concentrations, saturations, porosities, and methods that retrieve data from the workers back to the manager, such as concentrations, solution volumes, densities, and selected output.

An example of Fortran code that allows an MPI worker to participate in MPI calculations is shown in Table 8. The MPI worker creates a reaction module with RM\_Create. The number of user grid cells (*nxyz*) is not important for the workers because the manager sets its value when the worker PhreeqcRM instance is created. If the MPI process number (*mpi\_myself*) is greater than zero, the process is a worker, and RM\_MpiWorker is called, which is a loop that waits for a task message from the manager. The worker then processes tasks from the manager until the manager calls RM\_MpiWorkerBreak, which indicates that processing by the worker is complete. The worker returns from RM\_MpiWorker to destroy the worker’s reaction module and exit from the simulation.

Table 8: Example code for an MPI worker.

|  |
| --- |
| integer id, mpi\_myself, status  id = RM\_Create(nxyz, MPI\_COMM\_WORLD)  if (id < 0) then  stop "Failed to create reaction module"  endif  mpi\_myself = RM\_GetMpiMyself(id)  if (mpi\_myself > 0) then  status = RM\_MpiWorker(id);  status = RM\_Destroy(id);  return  endif |

PhreeqcRM has the algorithm described in PHAST [35] to balance the computational load among the number of threads or processes. A list of cells is divided into tasks, whereby the first cells in the list are assigned to the first task, the next cells are assigned to the second task, and so on, until the final cells are assigned to the th task, with the provision that . The load‑balancing algorithm adjusts the set of to try to achieve an equal calculation time for each of the tasks. Preliminary indications are that the algorithm works well for small numbers of tasks, but is not effective for large numbers of tasks on a cluster of computers. Methods to set the load balancing are RM\_SetRebalanceByCell and RM\_SetRebalanceFraction.

The developer can add tasks for the workers by using RM\_SetMpiWorkerCallback (Fortran) to register a function that will be called when a non‑PhreeqcRM task message (integer) is sent by the manager to RM\_MpiWorker. The registered method interprets the task messages and calls developer‑defined methods. PHAST uses a callback and additional non‑PhreeqcRM methods to distribute the transport simulations (one per component) among the available MPI processes. In PHAST, a callback function is registered; the manager sends a message to all workers that transport is to be calculated; the task message is not a PhreeqcRM task message, so the callback method is called; and the callback method calls a PHAST transport method that is run by the workers. Within the transport method, data is transferred from the manager to the workers, and the transport calculations are performed by specified workers. Other PHAST‑defined tasks, identified by task messages, are used to collect the post‑transport concentrations from the workers.

The speedup of parallelization, defined as fraction of CPU time compared to serial computations on a single processor, has been evaluated for the MoMaS easy benchmark (see section 3.4) by using the multithreaded (OpenMP) and the multiprocessing (MPI) versions of PhreeqcRM in PHAST. Two series of simulations using 1 to 16 threads or processes were run on a Linux computer with 24 cores (AMD Opteron 6168). The times for the chemistry calculation diminished from about 23 h to about 2 h for both multithreading and multiprocessing (Figure 2). However, in this limited testing on Linux, multiprocessing appears to be more efficient; it produced faster simulation times and greater speedups relative to multithreading. When using multiple nodes on a large multiprocessor computer with shared memory (Silicon Graphics, 256 CPUs), the MPI version was vastly superior to the OpenMP version. Apparently, the threads of the OpenMP version competed inefficiently to read and write shared memory. For the MPI version, the processes had independent data to run calculations, and the explicit distribution and collection of data was relatively more efficient than OpenMP.



Figure 2: Log-log plots of (a) chemistry calculation time (including communication) and (b) speedup against number of threads (Multithreaded) or number of processes (Multiprocessing) for the easy 1D MoMaS reactive transport benchmark.

# Code Verification

PhreeqcRM has been implemented as the reaction engine for the reactive transport calculations in PHAST [35] and FEFLOW [36]. Both simulators have been tested on a 3D reactive transport analytical solution [37] and a series of MoMaS reactive transport benchmarks [38].

## Implementation of PhreeqcRM as the Reaction Engine in PHAST

The changes to PHAST that implement PhreeqcRM are transparent to the user; input, output, and documentation are unchanged. The code that coupled the transport calculation (Fortran) to the reaction calculations was rewritten to use PhreeqcRM. The previous version of PHAST (version 2) relied on the source code of PHREEQC version 2 with additional coding to define initial and boundary conditions, transfer concentrations from transport to PhreeqcRM, run reaction calculations, and return concentrations for transport. PhreeqcRM was designed to perform these functions for PHAST, and the functions of the previous PHAST version correspond closely with the methods of PhreeqcRM.

PHAST’s finite-difference nodes are linked to reaction cells in PhreeqcRM by a mapping that accounts for inactive zones and removes redundancies based on symmetry. PHAST assumes constant fluid properties and ignores changes in density and fluid volume produced by reactions. Concentrations are transferred to the module before reaction calculations and retrieved from the module after reaction calculations.

New code was introduced to PHAST to retrieve the selected‑output data, and write files in the same format as the previous version, either as text or in HDF (Hierarchical Data Format). PHAST has another output format that is used to save solution definitions for a specified set of cells for use as boundary conditions in subsequent simulations on a more refined grid. To recreate this file format, concentrations are extracted from the concentration array used for transport calculations and transferred to the utility IPhreeqc instance to create PHREEQC solutions (Concentrations2Utility method). These solution definitions in the utility instance are then speciated and written to file by the RunString method using the PHREEQC keywords RUN\_CELLS and DUMP.

The other major coding effort for the new PHAST version involved parallelizing the transport calculations for each component. Fortran modules were written to contain all of the data necessary for a transport calculation, and methods were added to perform these calculations in parallel. For multithreading, a single loop was parallelized with OpenMP directives that assign component‑transport calculations to individual threads. For multiprocessing, PHAST makes use of the same processes used to parallelize the reaction calculations. To do the transport calculations with MPI, the method SetMpiWorkerCallbackFortran is called by each worker process to register a Fortran subroutine. This Fortran subroutine has a single integer argument, and calls other Fortran subroutines (depending on the argument value) that transfer data, perform transport calculations, and retrieve data from those processes that do the transport calculations. The manager process sends messages to the worker processes that determine the sequence of subroutines run by the workers.

## Implementation of PhreeqcRM as the Reaction Engine for FEFLOW

PhreeqcRM is implemented as an optional plugin for the groundwater modelling software FEFLOW [36] by using FEFLOW’s Interface Manager (IFM). In addition to the API, the IFM provides plugin methods that execute at certain events during program flow. Similar to PHAST, these event-based methods are implemented as callbacks for the FEFLOW main program. For the problem definition, result visualization, and result storage, FEFLOW’s existing infrastructure and GUI (Graphical User Interface) elements are used as much as possible.

Every node in FEFLOW’s finite-element mesh corresponds to a reaction cell in PhreeqcRM. Component concentrations, solution temperature and pressure, and liquid phase saturation are transferred from FEFLOW’s transport nodes to the reaction cells in PhreeqcRM before each reaction step. Saturation and density are transferred from PhreeqcRM to FEFLOW after each reaction step, which allows FEFLOW to simulate density-dependent flow as a function of temperature and chemical reactions.

The starting point of a coupled simulation is a fully functional FEFLOW model for flow and transport with at least one (placeholder) mass transport species. Transport properties of this species (e.g., diffusion coefficient, porosity) are used for all geochemical components. Geochemical components are determined from the associated PHREEQC input files and added automatically at the start of the simulation. Furthermore, the FEFLOW model requires a time series (*power curve* in FEFLOW terminology) that defines reaction steps and changes in boundary conditions and a named value distribution (*nodal user data* in FEFLOW terminology) that identifies nodes with the same geochemical initial conditions. Initial liquid phase saturation and temperatures are taken from the FEFLOW model.

In contrast to PHAST, the FEFLOW plugin uses separate PHREEQC input files to define the geochemical boundary and initial conditions. An IFM property editor is used to associate PHREEQC files with FEFLOW nodes with the same boundary and initial conditions (that is, the same value for the nodal user data).. For initial conditions, the solution with the highest user number in the file is transferred as the initial solution for all of the specified transport cells; any reactants with the same user number also are transferred.

For boundary conditions, the solution with the highest user number in each PHREEQC file is used to define the solution composition for all nodes with the same constant concentration boundary value. File associations and additional coupling settings are saved together with the FEFLOW *fem*‑file through the serialization functionality in the IFM.

Output from SELECTED\_OUTPUT and USER\_PUNCH keywords in the PHREEQC files that are used to define boundary and initial conditions is transferred to FEFLOW as an additional nodal user data definition, which is updated after each reaction step. Nodal user data definitions can be saved together with FEFLOW’s result files (*dac*‑ and *dar*‑files). This allows for the visualization of geochemical parameters during and after a coupled simulation using FEFLOW’s built-in post-processing tools.

## Kinetic Decay-Chain Test Case

The analytical solution of Wexler [42] in combination with the methodology for multi‑species transport of Sun et al. [37] is used to verify the calculation of simple kinetic reactions in PhreeqcRM and its implementation as the reaction engine for FEFLOW and PHAST. Results from the FEFLOW plugin are presented here. The example is adapted from Example 2 of the PHAST manual [43] and simulates a decay chain of four artificial species () according to the first order rate expressions

|  |  |
| --- | --- |
|  |  |

with species concentrations (mol/L) and rate constants (1/d) given by

|  |  |
| --- | --- |
|  |  |

The analytical solution is derived for a semi‑infinite half‑space with Cartesian coordinates where the x‑axis is chosen to be the normal vector of the boundary surface with the source patch at x = 0. Domain properties for the numerical approximation using the FEFLOW plugin are given in Table 9. Due to symmetry along the x‑y and x‑z planes, only the top left sector looking into the direction of flow is simulated. At the source patch, a constant concentration boundary with 0 mol/L is set for all species except species , which has a concentration of 1 mol/L. At all other boundaries, a zero‑gradient condition is employed for solute transport.

Table 9: Domain properties.

|  |  |
| --- | --- |
| Property | Value |
| Domain extent in x‑direction (m) | 100 |
| Domain extent in y‑direction (m) | 25 |
| Domain extent in z‑direction (m) | 15 |
| Flow velocity in x‑direction (m/d) | 2 |
| Longitudinal dispersivity (m) | 1.5 |
| Transverse dispersivity (m) | 0.15 |
| Source patch extent in y‑direction (m) | 0 to 5 |
| Source patch extent in z‑direction (m) | 0 to 2.5 |
| Simulation period (d) | 400 |
| Operator-splitting time step (d) | 10 |

Results in Figure 3 demonstrate excellent agreement between the FEFLOW plugin and the analytical solution. By using simulations with shorter operator-splitting time step, it can be shown that differences relative to the analytical solution result mainly from operator splitting. The operator-splitting error is most evident close to the source patch, where advection‑dominated inflow leads to an overestimation of concentrations of , while the rest of the decay-chain concentrations, , are underestimated. Minor differences of at the downstream boundary result from the approximation of the semi‑infinite half space by a finite domain of 100 m.

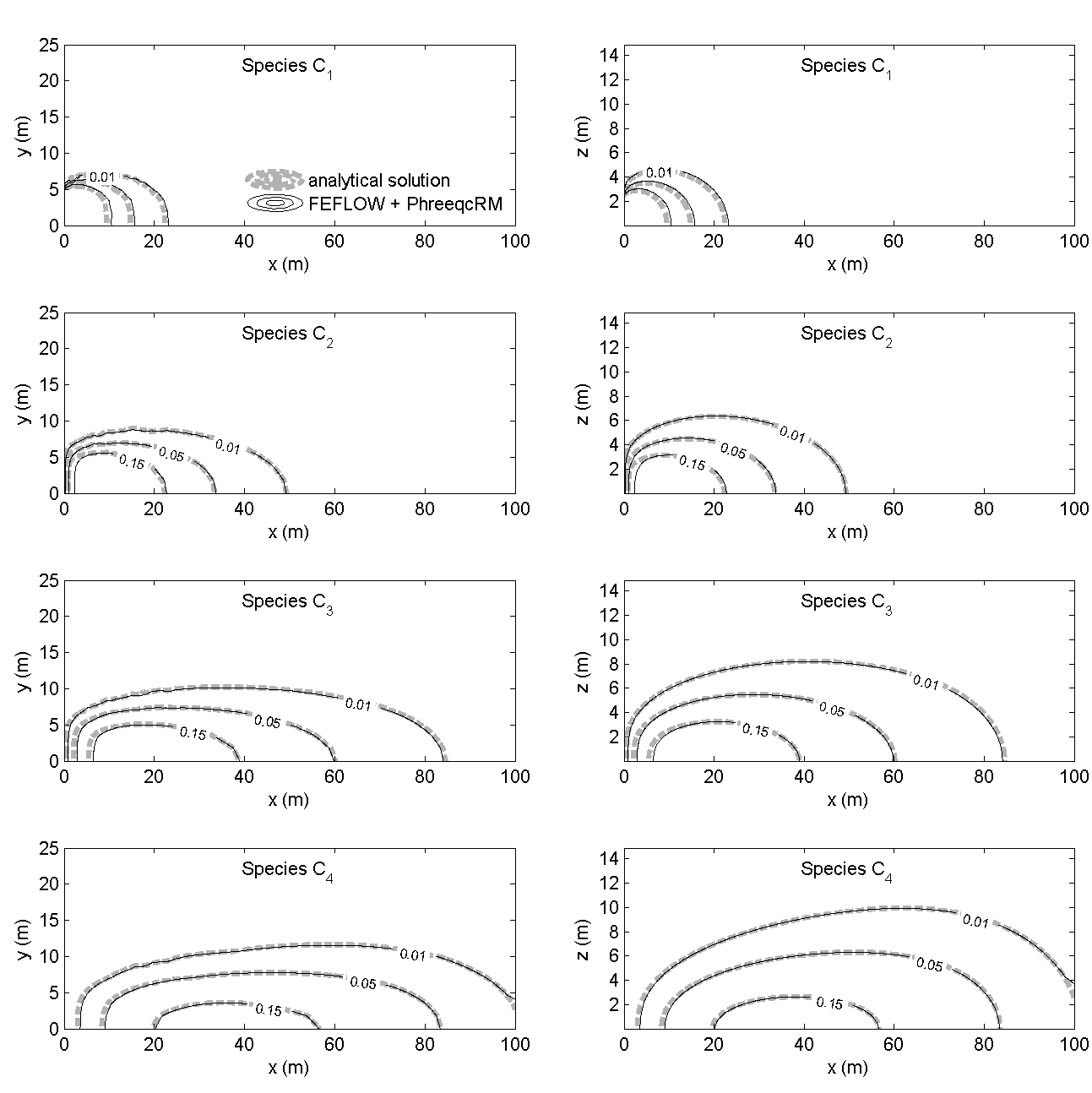


Figure 3: Species concentrations for a first-order decay chain in a steady flow field as calculated by the FEFLOW plugin compared to the analytical solution of Wexler [42] and Sun et al. [37].

## Reactive Transport Benchmarks of MoMaS

The MoMaS reactive transport benchmarks [38], referred to here as MoMaS, define test cases for steady-state flow with advection- or dispersion‑dominated transient solute transport in 1D and 2D domains together with three versions (easy, medium, and hard) of an artificial reaction network. The general applicability of the SNIA to the MoMaS exercise has been demonstrated for the SPECY code in Carrayrou et al. [44] and Carrayrou [45]. Details of the MoMaS definitions are provided by Carrayrou et al. [38] and Bourgeat et al. [46].

Results of the 1D cases with MPI timing are presented for PHAST, and results of the 2D cases with OpenMP timing are presented for the FEFLOW plugin. Because the MoMaS does not define units for dimensional properties, model input and results are presented in terms of unitless time, length, and concentration. Figures showing the complete set of 2D results are provided as additional material.

### Reaction Network

The hard test case defines equilibrium speciation reactions with five primary solution species and seven secondary solution species, surface complexation without electrical-double-layer calculations that involve one primary surface species and two secondary surface species, and two equilibrium and two kinetic reactions for solid phases. The medium and easy test cases are progressive simplifications of the reaction network from the hard case.

The implementation of the reaction networks in PHREEQC is complicated by the fact that PHREEQC requires positive concentrations of components and balanced chemical reactions, whereas the MoMaS reaction network has components that may have negative concentrations and partial mole-balance definitions. However, balanced chemical reactions can be obtained for PHREEQC by using specific definitions of species charge and associated hydrogen ions as shown in Table 10. The negative MoMaS component concentrations can be calculated by the appropriate combination of PHREEQC component and species concentrations.

Table 10: Correspondence between MoMaS and PHREEQC chemical formulas.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Reactant name | MoMaS | PHREEQC |
| Aqueous species |  | X1 | [X1] |
|  |  | X2 | H+ |
|  |  | X3 | [X3]-3 |
|  |  | X4 | H4[X4] +3 |
|  |  | X5 | [X5]- |
| Surface species |  | S | SOH, BOH,  depending on site concentration |
| Equilibrium phases | CP1 | X23X3 | H3[X3] |
|  | CP2 | X2X5 | H[X5] |
| Kinetic reactants | Cc | X2X4 | H[X4] gain to solution |
|  | X5 decomposition | X23X3  X2X5 | H3[X3] gain to solution  H[X5] loss from solution |

Because MoMaS reaction networks ignore activity correction, activity coefficients in PHREEQC were effectively set to unity by defining the parameters and of the extended Debye‑Hückel equation to 1012 and 0 respectively [47,48]. The activity of water was assumed to be 1.0 in the MoMaS, so the mass-action expression for OH- was adjusted to remove the effects of nonunity activity of water.

MoMaS defines mass-action equations for bidentate surface complexation in terms of molarity, which has been recognized as problematic [49], whereas PHREEQC uses the mole fraction of sites occupied for activity of surface species. To be consistent with the MoMaS definitions, it was necessary to define different equilibrium constants for different surface site concentrations. Thus, two surfaces were defined that had different equilibrium constants for the surface species; each was assigned to correspond with the surface site concentrations for the two subdomains of the 1D and 2D MoMaS problems. All definitions used standard PHREEQC input; no modification of the PhreeqcRM code was necessary.

### 1D MoMaS Calculations with PHAST

The 1D easy, advective benchmark requires a fine spatial discretization and small time steps to resolve a small concentration peak in a surface species at early times. Only the advective benchmarks are presented because the dispersive benchmarks are less restrictive in terms of spatial and temporal discretization. PHAST was discretized with 1760 nodes for comparison with Mayer et al. [50] and a 0.05 unit time step (Courant number 0.92) to provide adequate resolution of the surface-species concentration peak. The time step was maintained for the entire 6000-unit simulation time in part because PHAST lacks an automatic time-stepping algorithm, but also to provide a well-defined number of cell calculations that does not depend on variable time stepping. The total number of cell calculations for each simulation was approximately 2.1x108. Backward-in-time and upstream-in-space weighting, which is unconditionally stable but introduces some numerical dispersion, was used for all simulations. Estimates of the numerical dispersivities (in spatial units) due to space and time are 0.60x10-3 and 0.55x10‑3 , respectively, the sum of which is considerably less than the specified dispersivity, 0.1x10‑1. Operator-splitting dispersivity is not estimated, but, based on spreading in the results compared to results for smaller time steps, may be of the same order of magnitude as the specified dispersivity.

Figure 4 shows the resolution of the surface-species peak for 0.05-unit and 0.01-unit time step runs. The smaller time step produces a sharper peak and a deeper valley at distances beyond the peak, indicating that grid and time convergence is not completely achieved. Although, smaller time steps for this simulation are feasible, a better approach is to implement an automatic time-stepping algorithm [50], which would allow for variable time steps that achieve a specified accuracy. Although the height of the peak is similar in all calculations, the location of the peak is at approximately 0.01 distance in the PHAST calculations, whereas the location is closer to 0.02 in the Mayer et al. [50] and Amir and Kern [51] results. The reason for this small difference in location is not known. All other concentration results, which are resolved at a coarser scale, are consistent between PHAST and the other simulators.



Figure 4: Concentration of the primary surface species, S, at 10 units time as calculated with 0.05- and 0.01-unit time steps for the first 0.06 distance units from the inflow boundary and the entire domain (insert).

PHAST calculations were run on a heterogeneous cluster of Intel computers using the Linux operating system. One computer of lower speed was included in all calculations; for this computer, one CPU unit (as determined by the hardware test in the definition of MoMaS [38]) equals to 8.3 seconds; however, other computers used in the calculations were up to 25 percent faster.

A series of runs was made for the easy, medium, and hard benchmarks using 16 to 256 MPI processes and the SLURM (Simple Linux Utility for Resource Management) job control software. Some adjustments of SLURM and MPI parameters were made to avoid slower processors of the cluster, and to assign only one process per CPU core of each computer. With these adjustments, relatively consistent timings were obtained for the chemistry calculations, including the communication time to send and receive data between the MPI processes and the manager process (Figure 5).



Figure 5: (a) Normalized reaction-calculation times for PhreeqcRM, including communication time, for the easy, medium, and hard MoMaS 1D-advective benchmarks, and (b) speedup relative to a 16-process base case, as a function of number of MPI processes.

The addition of more processes resulted in faster calculations times for all three benchmarks. However, the relative speedup between 128 and 256 processes for the easy and medium test cases was small. The hard test case, as intended by the design of the MoMaS, required more computer time per cell, which resulted in a better speedup at 256 processes relative to the easy and medium test cases. Another consideration in the speedup results is that PhreeqcRM distributes the calculation load by assigning a number of cells to each process. If there are many cells and few processes, a difference of one cell among the processors may have a small effect. Whereas, with 256 processes and only 1760 reaction cells, the number of cells per process is about seven; even a difference of one cell results in about a 15 percent difference in load for a processor (assuming all cell calculations are equal).

The MoMaS 1D problem has relatively few cells (1760) and relatively few data items defined for selected output (22), although data are retrieved and written to file often (6000 times). If more data are retrieved from PhreeqcRM or larger output files are written, the communication times will increase relative to the reaction-calculation time, resulting in a decrease in the overall speedup. Finally, the heterogeneous nature of the cluster makes the timing results qualitative because the runs with more processes tend to use more of the slower processor cores. The calculation times and speedups would probably be better at high numbers of processes on a homogeneous cluster.

### 2D MoMaS Calculations with FEFLOW

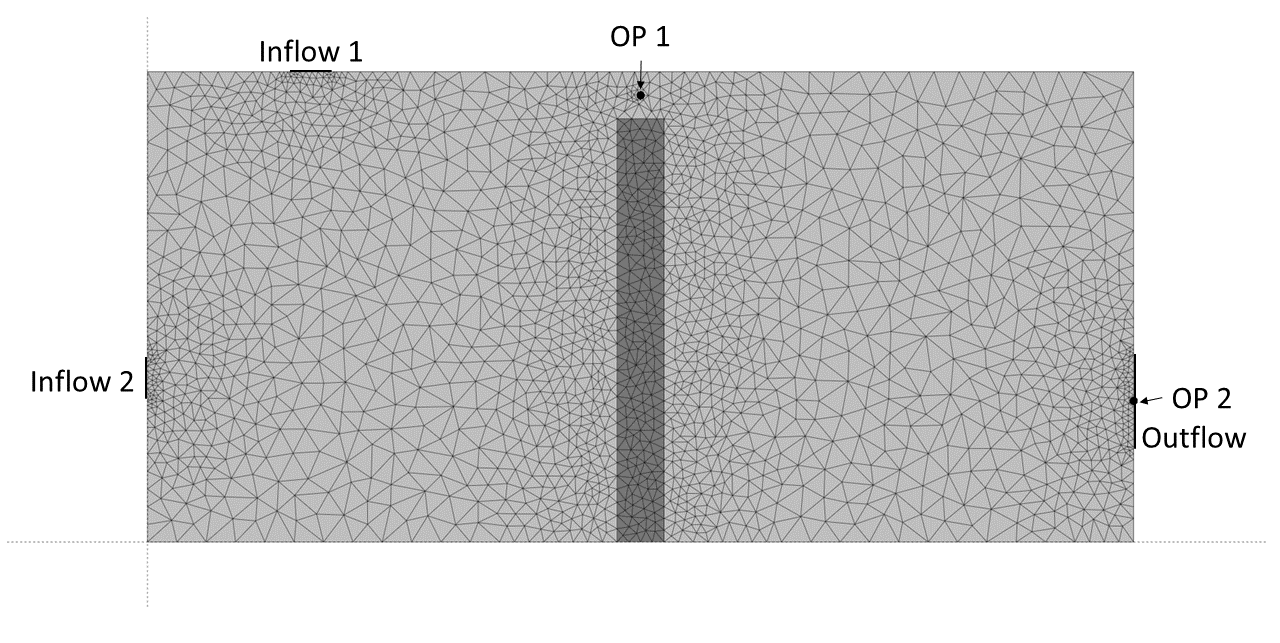


Figure 6: Finite-element mesh for the MoMaS 2D cases with high flow (light gray) and low flow (dark gray) subdomains, inflow and outflow zones, and observation points (OP).

The irregular finite-element mesh with 3753 elements and 1970 nodes used by the FEFLOW plugin for the 2D simulations is displayed in Figure 6. For the advective cases, reactions were calculated with an operator‑splitting time step of 1 for 100 time units at the beginning of the simulation and after the boundary change at time 5000; otherwise, a time step of 5 time units was used. Despite the suggestion to use the maximum Courant number as an indicator for the appropriate time step with the SNIA [44], the specified time stepping was chosen because FEFLOW uses a fully implicit method for non‑reactive solute transport, and the stability of the method is not limited by the Courant condition. Because of the greater transport fluxes of the dispersive cases, these cases were simulated with an operator-splitting time step of 0.1 for the first 20 time units of the simulation and after the boundary change; otherwise, an operator-splitting time step of 1 time unit was used. As an indicator of grid convergence, additional runs were conducted for a refined mesh consisting of 14930 elements and 7651 nodes with the same operator-splitting time step as the dispersive cases.

The streamline upwinding option was selected for the stabilization of solute transport. For the solution of the non-symmetric matrices from the transport equations, a standard iterative solver was used for the advective cases, and the direct, more stable but slightly slower PARDISO solver was used for the dispersive cases. A maximum error norm of 10‑4 was employed as an overall convergence criterion for flow and transport.

For the computer used for the calculations (Intel® Core™ i7‑4770 CPU @ 3.40GHz, 16 GB RAM), one CPU unit [38] equals to 7.0 seconds. Reaction calculations made use of OpenMP parallelization with eight threads. Timing of the 2D cases in CPU units is presented in Table 11.

Table 11: Calculation times for MoMaS in CPU units.

|  |  |  |  |
| --- | --- | --- | --- |
| Reaction network | Transport type | 2D FEFLOW | 2D fine FEFLOW |
| Easy | advective | 247 | 1730 |
| dispersive | 563 | 1628 |
| Medium | advective | 318 | 2039 |
| dispersive | 743 | 1937 |
| Hard | advective | 711 | 5839 |
| dispersive | 3171 | 9297 |

The implementation of PhreeqcRM in FEFLOW is not as fast as other published reactive transport codes [e.g., 50]. The performance of the coupling is hampered by the automated time stepping algorithm (predictor-corrector) in the implicit solution of solute transport. In the coupling to PhreeqcRM, the algorithm greatly reduces the transport time step after each reaction step due to abrupt changes and discontinuities in the concentration fields that result from the chemical reactions.



Figure 7: MoMaS results from 2D advective cases: (a) X3, easy case at time 1000; (b) X2, medium case at time 1000; (c) CP1, hard case at time 2000; (d) same as (c) calculated with refined mesh.

Selected results of the 2D advective MoMaS cases are presented in Figure 7. According to Carrayrou et al. [44] the concentration of component X3 at time 1000 offers a suitable means for comparison. Plotted with a similar color map, Figure 7a can be directly compared to their Fig. 8. Given the bandwidth of results in Carrayrou et al. [44], results from FEFLOW plugin fit well with results from MIN3P [52,53], Hoffmann et al. [54,55] and HYTEC [56,57]. Apart from slightly higher concentrations, the simulation with the refined mesh (not shown) yields a very similar concentration pattern.

Figure 7b shows the concentration of X2 with a similar color map as in Mayer and MacQuarrie [50] to facilitate direct comparison to their Fig. 6c. Simulations with the refined mesh (not shown) yield a smoother reaction edge but otherwise the same general concentration pattern.

Figure 7c shows the equilibrium mineral species CP1 at time 2000 for the hard advective case. Qualitatively, results are similar to those in Mayer and MacQuarrie [50], Fig. 7g , however, maximum concentrations are lower (~20). Using the refined mesh, Figure 7d reveals the discretization dependence of maximum CP1 concentrations. With the refined mesh, results are also quantitatively comparable to Mayer and MacQuarrie [50]. In addition to the figures presented here, all results that are specified by the MoMaS definition [38] are similar to results from Hoffmann [55] using RICHY2D.



Figure 8: FEFLOW results for the MoMaS 2D dispersive cases; (a) and (b) S, easy case at time 10 with coarse and fine mesh, respectively; (c) X2, medium case at time 10; (d) CP1, hard case at time 2000.

Figure 8 verifies the results from the dispersive cases relative to published data in Carrayrou et al. [44] and Mayer and MacQuarrie [50]. For comparability with Carrayrou et al. [44], Fig. 11, a similar color scale is used in Figure 8a and b. Results obtained with the finer mesh show only minor differences compared to results for the coarser mesh (Figure 8b versus Figure 8a). Compared to results from Mayer and MacQuarrie [50], Fig. 6d, component X2 at time 10 (Figure 8c) has a sharper transition from low to high concentrations and has less lateral spread at Inflow 2. Compared to their Fig. 7h, the mineral phase CP1 at time 2000 (Figure 8d) is removed from a larger part of the domain. Furthermore, the large amounts of this phase at the edge of the low-permeability zone is not reproduced. Results with the operator-splitting time step of the coarse-meshed advective case show significant differences compared to the presented results. These differences indicate the need for comparative operator-splitting time step and mesh convergence studies with the SNIA in order to demonstrate the reliability of results.

# Summary

PhreeqcRM is a module based on the geochemical model PHREEQC that is designed to perform reaction calculations for reactive transport simulators that use an operator‑splitting approach. This reaction module relies on IPhreeqc, the previous encapsulation of PHREEQC, but specializes its use specifically for reactive transport. The module has methods to set initial conditions, set boundary conditions, transfer concentrations and other model properties, run reactions, and retrieve post-reaction results. The module can provide component concentrations for simulators that use species‑independent transport parameters or aqueous species concentrations for multicomponent-diffusion simulators. PhreeqcRM has been implemented in two simulators, PHAST and FEFLOW, which have been used to demonstrate the validity and efficiency of the approach on a test problem with an analytical solution and on the MoMaS reactive transport benchmarks. PhreeqcRM is parallelized for OpenMP or MPI, depending on compilation options, and good scalability has been demonstrated on multiprocessor computers with up to 16 cores and on a multicomputer cluster with up to 256 cores. PhreeqcRM is written in C++, but interfaces allow methods to be called from C or Fortran90. By using the PhreeqcRM reaction module, an existing multicomponent transport simulator can be extended to simulate a wide range of geochemical reactions.

# Acknowledgments

We thank Jeffrey Falgout for his help in using the U.S. Geological Survey computer cluster (arc1) in Lakewood, Colorado. The contribution of Laurin Wissmeier was partially funded by AF Consult Switzerland Ltd.

# References

[1] Zhang GX, Spycher N, Sonnenthal E, Steefel C, Xu TF. Modeling reactive multiphase flow and transport of concentrated solutions. Nucl Technol 2008;164:180–95.

[2] Kletskova T, Czerwinski K, Gelbard E, Yip S. Modeling cation exchange in zeolitic nuclear waste form. Journal of Computer-Aided Materials Design 1999;6:363–8.

[3] Cochepin B, Trotignon L, Bildstein O, Steefel CI, Lagneau V, van der Lee J. Approaches to modelling coupled flow and reaction in a 2D cementation experiment. Adv Water Resour 2008;31:1540–51. doi:10.1016/j.advwatres.2008.05.007.

[4] Arnold J, Kosson DS, Garrabrants A, Meeussen JCL, van der Sloot HA. Solution of the nonlinear Poisson–Boltzmann equation: Application to ionic diffusion in cementitious materials. Cement and Concrete Research 2013;44:8–17. doi:10.1016/j.cemconres.2012.10.013.

[5] Appelo CAJ, van Loon LR, Wersin P. Multicomponent diffusion of a suite of tracers (HTO, Cl, Br, I, Na, Sr, Cs) in asingle sample of Opalinus Clay. Geochim Cosmochim Ac 2010;74:1201–19. doi:10.1016/j.gca.2009.11.013.

[6] Appelo CAJ, Vinsot A, Mettler S, Wechner S. Obtaining the porewater composition of a clay rock by modeling the in- and out-diffusion of anions and cations from an in-situ experiment. J Contam Hydrol 2008;101:67–76. doi:10.1016/j.jconhyd.2008.07.009.

[7] Anthony C, Appelo CAJ, Wersin P. Multicomponent diffusion modeling in clay systems with application to the diffusion of tritium, iodide, and sodium in Opalinus clay. Environ Sci Technol 2007;41:5002–7. doi:10.1021/Es0629256.

[8] Trotignon L, Devallois V, Peycelon H, Tiffreau C, Bourbon X. Predicting the long term durability of concrete engineered barriers in a geological repository for radioactive waste. Phys Chem Earth Phys Chem Earth 2007;32:259–74.

[9] Montarnal P, Mügler C, Colin J, Descostes M, Dimier A, Jacquot E. Presentation and use of a reactive transport code in porous media. Physics and Chemistry of the Earth, Parts A/B/C 2007;32:507–17. doi:10.1016/j.pce.2006.01.009.

[10] Steefel CI, Lasaga AC. A coupled model for transport of multiple chemical species and kinetic precipitation dissolution reactions with application to reactive flow in single-phase hydrothermal systems. American Journal of Science 1994;294:529–92.

[11] Oldenburg CM, Pruess K. Simulation of propagating fronts in geothermal reservoirs with the implicit Leonard total variation diminishing scheme. Geothermics 2000;29:1–25.

[12] Lichtner PC. Continuum model for simultaneous chemical-reactions and mass-transport in hydrothermal systems. Geochimica Et Cosmochimica Acta 1985;49:779–800.

[13] Bozau E, van Berk W. Hydrogeochemical Modeling of Deep Formation Water Applied to Geothermal Energy Production. Proceedings of the Fourteenth International Symposium on Water-Rock Interaction, WRI 14, vol. 7, 2013, p. 97–100. doi:10.1016/j.proeps.2013.03.006.

[14] Wissmeier L, Barry DA, Phillips IR. Predictive hydrogeochemical modelling of bauxite residue sand in field conditions. Journal of Hazardous Materials 2011;191:306–24. doi:10.1016/j.jhazmat.2011.04.078.

[15] Liu WC, Yang JK, Xiao B. Review on treatment and utilization of bauxite residues in China. International Journal of Mineral Processing 2009;93:220–31. doi:10.1016/j.minpro.2009.08.005.

[16] Xenidis A, Harokopou AD, Mylona E, Brofas G. Modifying alumina red mud to support a revegetation cover. Jom 2005;57:42–6.

[17] Jurjovec J, Blowes DW, Ptacek CJ, Mayer KU. Multicomponent reactive transport modeling of acid neutralization reactions in mine tailings. Water Resources Research 2004;40. doi:10.1029/2003wr002233.

[18] Amos RT, Mayer KU, Blowes DW, Ptacek CJ. Reactive transport modeling of column experiments for the remediation of acid mine drainage. Environmental Science & Technology 2004;38:3131–8. doi:10.1021/es0349608.

[19] Xu T, Li J. Reactive Transport Modeling to Address the Issue of CO2 Geological Sequestration. Proceedings of the Fourteenth International Symposium on Water-Rock Interaction, WRI 14, vol. 7, 2013, p. 912–5. doi:10.1016/j.proeps.2013.03.153.

[20] Graupner BJ, Li D, Bauer S. The coupled simulator ECLIPSE–OpenGeoSys for the simulation of CO2 storage in saline formations. Energy Procedia 2011;4:3794–800. doi:10.1016/j.egypro.2011.02.314.

[21] Zhang YQ, Oldenburg CM, Finsterle S, Bodvarsson GS. System-level modeling for economic evaluation of geological CO2 storage in gas reservoirs. Energ Convers Manage Energ Convers Manage 2007;48:1827–33.

[22] Xu TF, Sonnenthal E, Spycher N, Pruess K. TOUGHREACT - A simulation program for non-isothermal multiphase reactive geochemical transport in variably saturated geologic media: Applications to geothermal injectivity and CO2 geological sequestration. Comput Geosci-Uk 2006;32:145–65. doi:10.1016/j.cageo.2005.06.014.

[23] Nordbotten JM, Kavetski D, Celia MA, Bachu S. Model for CO2 leakage including multiple geological layers and multiple leaky wells. Environ Sci Technol Environ Sci Technol 2009;43:743–9.

[24] Kvamme B, Liu S. Reactive transport of CO2 in saline aquifers with implicit geomechanical analysis. Energy Procedia 2009;1:3267–74. doi:10.1016/j.egypro.2009.02.112.

[25] Meeussen JCL. ORCHESTRA: An object-oriented framework for implementing chemical equilibrium models. Environmental Science & Technology 2003;37:1175–82. doi:10.1021/es.025597s.

[26] Van der Lee J. Thermodynamic and mathematical concepts of CHESS. Fontainebleau, France.: Ecole des Mines de Paris; 1998.

[27] Charlton SR, Parkhurst DL. Modules based on the geochemical model PHREEQC for use in scripting and programming languages. Computers & Geosciences 2011;37:1653–63. doi:10.1016/j.cageo.2011.02.005.

[28] David L. Parkhurst, C.A.J. Appelo. Description of Input and Examples for PHREEQC Version 3—A Computer Program for Speciation, Batch-Reaction, One-Dimensional Transport, and Inverse Geochemical Calculations. Denver, Colorado: U.S. Department of the Interior, U.S. Geological Survey; 2013.

[29] Wissmeier L, Barry DA. Simulation tool for variably saturated flow with comprehensive geochemical reactions in two- and three-dimensional domains. Environ Modell Softw 2011;26:210–8. doi:10.1016/j.envsoft.2010.07.005.

[30] Nardi A, Idiart A, Trinchero P, de Vries LM, Molinero J. Interface COMSOL-PHREEQC (iCP), an efficient numerical framework for the solution of coupled multiphysics and geochemistry. Computers & Geosciences 2014;69:10–21. doi:10.1016/j.cageo.2014.04.011.

[31] Kolditz O, Bauer S, Bilke L, Böttcher N, Delfs JO, Fischer T, et al. OpenGeoSys: an open-source initiative for numerical simulation of thermo-hydro-mechanical/chemical (THM/C) processes in porous media. Environ Earth Sci 2012;67:589–99. doi:10.1007/s12665-012-1546-x.

[32] Takahashi Y, Ishida T. Modeling of coupled mass transport and chemical equilibrium in cement-solidified soil contaminated with heavy-metal ions. Construction and Building Materials n.d. doi:10.1016/j.conbuildmat.2013.10.030.

[33] Huber P, Nivelon S, Ottenio P, Nortier P. Coupling a Chemical Reaction Engine with a Mass Flow Balance Process Simulation for Scaling Management in Papermaking Process Waters. Ind Eng Chem Res 2012;52:421–9. doi:10.1021/ie300984y.

[34] Patel RA, Perko J, Jacques D, Schutter GD, Breugel KV, Ye G. A versatile pore-scale multicomponent reactive transport approach based on Lattice Boltzmann Method: Application to portlandite dissolution. Physics and Chemistry of the Earth, Parts A/B/C n.d. doi:10.1016/j.pce.2014.03.001.

[35] Parkhurst DL, Kipp KL, Charlton SR. PHAST Version 2—A program for simulating groundwater flow, solute transport, and multicomponent geochemical reactions. U.S. Geological Survey; 2010.

[36] DHI-WASY. FEFLOW® 6.2 User Manual 2014.

[37] Sun Y, Petersen JN, Clement TP. Analytical solutions for multiple species reactive transport in multiple dimensions. Journal of Contaminant Hydrology 1999;35:429–40. doi:10.1016/S0169-7722(98)00105-3.

[38] Carrayrou J, Kern M, Knabner P. Reactive transport benchmark of MoMaS. Computational Geosciences 2010;14:385–92. doi:10.1007/s10596-009-9157-7.

[39] Barry DA, Bajracharya K, Crapper M, Prommer H, Cunningham CJ. Comparison of split-operator methods for solving coupled chemical non-equilibrium reaction/groundwater transport models. Mathematics and Computers in Simulation 2000;53:113–27.

[40] Carrayrou J, Mose R, Behra P. Operator-splitting procedures for reactive transport and comparison of mass balance errors. Journal of Contaminant Hydrology 2004;68:239–68.

[41] Yeh GT, Tripathi VS. A critical evaluation of recent developments in hydrogeochemical transport models of reactive multichemical components. Water Resour Res 1989;25:93–108. doi:10.1029/WR025i001p00093.

[42] Wexler EJ. Analytical solutions for one-, two-, and three-dimensional solute transport in ground-water systems with uniform flow, 1992, p. 190.

[43] Parkhurst DL, Kipp KL, Engesgaard P, Charlton SR. PHAST - A Program for simulating ground-water flow, solute transport, and multicomponent geochemical reactions. Denver, Colorado: U.S. Geological Survey; 2004.

[44] Carrayrou J, Hoffmann J, Knabner P, Kräutle S, de Dieuleveult C, Erhel J, et al. Comparison of numerical methods for simulating strongly nonlinear and heterogeneous reactive transport problems—the MoMaS benchmark case. Comput Geosci 2010;14:483–502. doi:10.1007/s10596-010-9178-2.

[45] Carrayrou J. Looking for some reference solutions for the reactive transport benchmark of MoMaS with SPECY. Comput Geosci 2010;14:393–403. doi:10.1007/s10596-009-9161-y.

[46] A. Bourgeat, S. Bryant, J. Carrayrou, A. Dimier, C.J. Van Duijn, M. Kern, et al. GdR MoMaS. Centre National del la Recherche Scientifique; 2008.

[47] Parkhurst DL, Appelo CAJ. User’s guide to PHREEQC (version 3)—A Computer Program for Speciation, Batch-Reaction, One-Dimensional Transport, and Inverse Geochemical Calculations. U.S. Geological Survey Techniques and Methods, vol. 6, chap. A43, n.d., p. 497.

[48] Parkhurst DL, Appelo CAJ. User’s guide to PHREEQC (version 2): A computer program for speciation, batch-reaction, one-dimensional transport, and inverse geochemical calculations. Denver, Colorado: U.S. Geological Survey; 1999.

[49] Wang Z, Giammar DE. Mass Action Expressions for Bidentate Adsorption in Surface Complexation Modeling: Theory and Practice. Environ Sci Technol 2013;47:3982–96. doi:10.1021/es305180e.

[50] Mayer K, MacQuarrie K. Solution of the MoMaS reactive transport benchmark with MIN3P—model formulation and simulation results. Computational Geosciences 2009. doi:10.1007/s10596-009-9158-6.

[51] Amir L, Kern M. A global method for coupling transport with chemistry in heterogeneous porous media. Comput Geosci 2010;14:465–81. doi:10.1007/s10596-009-9162-x.

[52] Mayer KU. A numerical model for multicomponent reactive transport in variably saturated porous media. Ph.D. University of Waterloo, Canada, 1999.

[53] Mayer KU, Frind EO, Blowes DW. Multicomponent reactive transport modeling in variably saturated porous media using a generalized formulation for kinetically controlled reactions. Water Resources Research 2002;38:131–1321.

[54] Hoffmann J, Kräutle S, Knabner P. A parallel global-implicit 2-D solver for reactive transport problems in porous media based on a reduction scheme and its application to the MoMaS benchmark problem. Comput Geosci 2010;14:421–33. doi:10.1007/s10596-009-9173-7.

[55] J. Hoffmann. Results of the GdR MoMaS Reactive Transport Benchmark with RICHY2D. Erlangen, Germany: Department of Mathematics, Section Modeling, Simulation, Optimization, University of Erlangen-Nuremberg; n.d.

[56] Van der Lee J, De Windt L, Lagneau V, Goblet P. Presentation and application of the reactive transport code HYTEC. In: S. Majid Hassanizadeh RJS, William G. Gray and George F. Pinder, editor. Developments in Water Science, vol. Volume 47, Elsevier; 2002, p. 599–606.

[57] Lagneau V, van der Lee J. HYTEC results of the MoMas reactive transport benchmark. Comput Geosci 2010;14:435–49. doi:10.1007/s10596-009-9159-5.