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 reaction module, based on the geochemical model PHREEQC, 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 simulation from the reaction simulations. The basic function of the reaction module is to take component concentrations for the model cells from the transport simulation, run geochemical reactions, and return revised concentrations. If multicomponent diffusion is modeled (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, bacterial cell growth, or any other kinetic process. PhreeqcRM assigns initial and boundary conditions for model cells based on standard PHREEQC 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 reaction module uses parallel processing to speed the reaction calculations, either by multi-threading with OpenMP on shared memory systems, or by 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 model can be easily extended to simulate a wide range of geochemical reactions.

Keywords:

Reactive transport, PHREEQC, PHAST, FEFLOW, modelling, reaction engine, parallelization, component transport, species transport, multi-component diffusion

# 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 at the center of interest because of (1) its potential to predict the evolution of contaminant plumes over long time scales, and (2) its capabilities 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 strategies [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 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 has become 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. [\*\*add? Recently, Nardi et al. (2014) have published another COMSOL-IPhreeqc coupling.\*\*] The free environmental flow- and transport-modelling platform OpenGeoSys [30] has an interface to IPhreeqc for coupled thermo-hydro-mechanical-chemical (THMC) simulations. The HP1 module is a coupling of the popular software package HYDRUS for variably saturated flow and transport with PHREEQC for reaction calculations [31]. Specialized tools have been developed using IPhreeqc 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 using a lattice Boltzmann based approach. This non-comprehensive list indicates scientific and industrial interest in using PHREEQC as reaction module.

However, IPhreeqc is not specifically tailored for couplings with transport simulators. Its API is designed to provide access to all of PHREEQC’s reaction capabilities by reading and interpreting PHREEQC script commands from input strings or files. Albeit retaining flexibility, this approach limits the performance of data exchange and obstructs the readability of the client code that needs to compose the PHREEQC scripts according to the needs of the transport simulator.

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. PhreeqcRM provides an API where all properties that are relevant for couplings with transport simulators can be directly accessed without requiring PHREEQC scripting commands. 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.

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

# The PhreeqcRM API

PHREEQC is a general-purpose geochemical reaction model that has capabilities to model interactions between water 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 of reactions to consider and the compositions of the reactants. IPhreeqc [] encapsulated 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 relied primarily on strings that define keyword data blocks and processing arrays of selected output data to automate the use of PHREEQC. For reactive-transport modeling, 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 models. The methods of the class allow data arrays to be transferred and reaction calculation to be performed without the use of keyword data blocks. Selected-output data can be obtained as an array, but the intent is primarily for output to files. The methods in PhreeqcRM include all of the capabilities necessary to implement PHREEQC reaction calculations into a multicomponent transport model.

## Main interface functionality

static or dynamic linked library for Fortran, C and C++

Java wrapper using SWIG

Instead of going through PHREEQC’s keyword-based scripting language, geochemical properties can be directly accessed.

The use of PhreeqcRM consists of a sequence of method calls that create and initialize a PhreeqcRM 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 destroy the instance. Some of the key C++ methods that provide these functionalities are described in this section; however, an equivalent subroutine, usually the same name preceded by “RM\_”, is available for transport codes written in C and Fortran.

### Create and Initialize PhreeqcRM

PhreeqcRM is a C++ class, and an instance can be created simply by a declaration statement, or by using the C++ keyword “new.” In the example (table xxx), an instance named phreeqc\_rm is constructed assuming that the source code has been compiled for use with multiple threads (OPENMP). Three threads will be used for the calculations, and the transport model has 40 cells. In Fortran or C, the subroutine RM\_Create is used to create the instance. Table xxx has a sampling of other methods that may be called to initialize and set the operation of the reaction module.

The action taken when an error is encountered can be set to one of three options: the current method returns an error code, an exception is thrown, or the program aborts. A prefix may be defined for the names of the output (*prefix.chem.txt*) and log files (*prefix.log.txt*) by using the method SetFilePrefix.

The transport model 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 are several orders of magnitude smaller in concentration than the totals, 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 the H in water, total O minus the 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 (the default) is minimal.

The number of cells for reaction calculations may differ from the number of cells in the transport model because of inactive zones or symmetry—for example, if a finite-difference model is always three dimensional (3D), but the problem is logically two dimensional (2D), then it is possible to run reaction calculations on one plane of cells and copy the concentration results to other planes to produce the full 3D distribution of concentrations. 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 may be less than or equal to the number of transport cells (CreateMapping method). In general, PhreeqcRM returns results for every cell in the transport model, even if the number of reaction cells is less than the number of transport.

The reaction cells are defined to have a representative volume, , of 1 L. 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, and is porosity defined by

|  |  |
| --- | --- |
| , |  |

where (L) is the total pore volume of the cell, and (L) is the total volume of the cell. The saturation, cell volume, and cell pore volume are set with the methods SetSaturation, SetCellVolume, and SetPoreVolume, respectively.

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

, where *ci* is mg/L and *gi* is gram formula weight of species *i*,

, where *Mi* is mol/L, or

, where *fi* is mass fraction, and *rho* is the solution density (as set by SetDensity).

To convert from solid-phase moles in the PhreeqcRM input, (mol), to the moles of solid phases in the representative volume of the cell, (mol), three different conversions () are available:

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

With conversion , the solid-phase input in PhreeqcRM () is moles per representative volume; with conversion , is moles per pore volume (fully saturated water volume); and with conversion C, is moles per volume of solid. The conversion from moles of solid- and gas-phase input in a PHREEQC input file to moles in the cell is specified by the methods SetUnitsPPassemblage, SetUnitsExchange, SetUnitsSurface, SetUnitsGasPhase, SetUnitsSSassemblage, SetUnitsKinetics.

[LoadDatabase].

Table xxx:

int nxyz = 40;

PhreeqcRM phreeqc\_rm(nxyz, 3);

phreeqc\_rm.SetErrorHandlerMode(1);

phreeqc\_rm.SetComponentH2O(false);

phreeqc\_rm.SetFilePrefix("Advect\_cpp");

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, cell\_vol, pv;

sat.resize(nxyz, 1.0);

phreeqc\_rm.SetSaturation(sat);

cell\_vol.resize(nxyz, 1);

phreeqc\_rm.SetCellVolume(cell\_vol);

pv.resize(nxyz, 0.2);

phreeqc\_rm.SetPoreVolume(pv);

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);

phreeqc\_rm.LoadDatabase("phreeqc.dat");

### Initial Conditions and Reactions for Each Cell

Initial conditions for solution compositions and solid and gas reactants can be read from a PHREEQC input file (or string) and distributed to the reaction cells. The set of reactants assigned to each cell define the types of reactions that can occur in the cell and the initial number of moles of each reactant, for example the initial amount of calcite and the number of ion-exchange sites.

A reaction module has at least three IPhreeqc instances, one worker (or more, if using OPENMP), an input-processing instance called the initial-phreeqc instance, and a utility instance, which optionally can be used for special calculations. (If using MPI, each process has a reaction module with exactly three IPhreeqc instances.) The RunFile method is used to read and execute a PHREEQC input file by any combination of IPhreeqc instances. If only initial 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 are using the same thermodynamic data and aqueous model. The strategy for assigning initial conditions to the reaction cells is to transfer definitions from the InitialPhreeqc instance to the workers.

Once the InitialPhreeqc instance reads a PHREEQC input file, a set of solution and reactant entities are present within the InitialPhreeqc instance, each identified by an integer user number. These entities 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 entity number for each transport cell for each type of entity. The types of entities are 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 entity is not present in a cell. The mapping from transport cells to reaction cells is used to define the entities in the worker IPhreeqc instances with reaction-cell numbering.

An alternative form of the InitialPhreeqc2Module method defines two entities 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.

An alternative way to distribute initial conditions is to run a PHREEQC input file and select one user number to assign to a each cell in a list of transport cells. All of the entities with that user number will be distributed to each cell in the list. The mapping from transport cells to reaction cells is used to define the entities in the worker IPhreeqc instances with reaction-cell numbering. Multiple files are processed to distribute initial conditions to all of the reaction cells. 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. In most cases, this list is the set of components that need to be transported by the transport code. 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.

[initial equilibration]

[get concentrations]

Table xxx:

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);

### Boundary Conditions

[InitialPhreeqc2Concentrations]

Table xxx:

std::vector<double> bc\_conc, bc\_f1;

std::vector<int> bc1, bc2;

int nbound = 1;

bc1.resize(nbound, 0); // solution 0 from Initial IPhreeqc instance

bc2.resize(nbound, -1); // no bc2 solution for mixing

bc\_f1.resize(nbound, 1.0); // mixing fraction for bc1

phreeqc\_rm.InitialPhreeqc2Concentrations(bc\_conc, bc1, bc2, bc\_f1);

### Time Stepping

bool print\_selected\_output\_on = (steps == nsteps - 1) ? true : false;

bool print\_chemistry\_on = (steps == nsteps - 1) ? true : false;

phreeqc\_rm.SetSelectedOutputOn(print\_selected\_output\_on);

phreeqc\_rm.SetPrintChemistryOn(print\_chemistry\_on, false, false); phreeqc\_rm.SetPoreVolume(pv);

phreeqc\_rm.SetSaturation(sat);

phreeqc\_rm.SetTemperature(temperature);

phreeqc\_rm.SetPressure(pressure);

phreeqc\_rm.SetConcentrations(c);

phreeqc\_rm.SetTimeStep(time\_step);

time += time\_step;

status = phreeqc\_rm.SetTime(time);

status = phreeqc\_rm.RunCells();

phreeqc\_rm.GetConcentrations(c);

std::vector<double> density;

phreeqc\_rm.GetDensity(density);

const std::vector<double> &volume = phreeqc\_rm.GetSolutionVolume();

### Destroy PhreeqcRM

### Single input file

### Multiple input files

### Multi-component flow

### Multi-species flow

### Mapping of transport nodes and chemistry cells

### Input conversion

The reaction cells are defined to have a representative volume, , of 1 L. A solution defined in PhreeqcRM input is transferred to the representative volume of a model cell in proportion to the water content, (L), according to

|  |  |
| --- | --- |
| , |  |

where is the liquid phase saturation, and is porosity defined by

|  |  |
| --- | --- |
| , |  |

where (L) is the total pore volume of the cell, and (L) is the total volume of the cell.

To convert from solid-phase properties in the PhreeqcRM input, (mol), to the moles of solid phases in the representative volume of the cell, (mol), three different conversions () are available:

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

With conversion , the solid-phase input in PhreeqcRM () is moles per representative volume; with conversion , is moles per saturated water volume; and with conversion C, is moles per volume of solid.

## Host application requirements

## 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 each cell can be distributed easily to multiple processors by using multiple threads, on a shared memory system, or multiple processes, on a distributed memory system. By using preprocessor definitions (USE\_OPENMP or USE\_MPI), PhreeqcRM can be compiled to use OpenMP, which enables multiple threads, or MPI, which enables multiple processes. The number of threads used by OPENMP is one of the arguments when creating a PhreeqcRM instance. The number of processes and the set of host computers used by MPI are defined through arguments to *mpiexec*, thecommand that launches an MPI job.

The implementation of multiple threads is relatively straightforward because 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 the reaction calculations on shared memory computers.

The parallelization for multiple processes is more complicated because all of the data necessary to perform a set of reaction calculations must be distributed to each process. 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 manager (root) process is the client, and the non-manager processes are the servers (workers). For example, when the method RunCells is run, 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 servers then wait for the next task message in MpiWorker This same server-client 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 other cell and module properties, and that retrieve data from the workers back to the manager, such as concentrations, solution volumes, densities, and other cell and module properties.

An example of Fortran code that allows an MPI worker to participate in MPI calculations is shown in figure/table xxx. The MPI worker creates a reaction module with RM\_Create. The number of user grid cells (*nxyz*) is not important for the workers because its value is set by the manager 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 xxx:

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 [] to balance the computational load among the number of threads or processes (*n*). A list of *M* cells is divided into *n* tasks, whereby the first *m1* cells in the list are assigned to the first task, the next *m2* cells are assigned to the second task, and so on, until the final *mn*cells are assigned to the *n*th task, with the provision that . The load-balancing algorithm adjusts the set of *mi* 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 a large numbers of tasks on a cluster of computers.

The developer can add tasks for the workers by using RM\_SetMpiWorkerCallback (Fortran) to register a method 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.



## Generic program flow

First set initial porosity and saturation, then transfer file data?

# Implementation as reaction engine for PHAST and FEFLOW

## Verification

### Chain of four kinetically decaying reactants

### MoMaS reactive-transport benchmark

# Conclusions

# Acknowledgments

# 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.

[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.

[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.

[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.

[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.

[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.

[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. Procedia Earth and Planetary Science 2013;7:97–100.

[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.

[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.

[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.

[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.

[19] Xu T, Li J. Reactive Transport Modeling to Address the Issue of CO2 Geological Sequestration. Procedia Earth and Planetary Science 2013;7:912–5.

[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.

[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.

[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.

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

[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.

[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.

[30] 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.

[31] Šimůnek J, Jacques D, van Genuchten MT, Mallants D. Multicomponent geochemical transport modeling using HYDRUS-1D and HP1. Journal of the American Water Resources Association 2006;42:1537–47.

[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.

[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.

[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.

[] Sun, Y., J. N. Petersen, and T. P. Clement. "Analytical solutions for multiple species reactive transport in multiple dimensions." *Journal of Contaminant Hydrology* 35.4 (1999): 429-440.

[] Centre National del la Recherche Scientifique, 2008, 18 p. https://www.ljll.math.upmc.fr/cances/gdrmomas/Ex\_qualif/Geochimie/Documents/Benchmark-MoMAS.pdf.

[] Centre National del la Recherche Scientifique, 2008, 18 p. https://www.ljll.math.upmc.fr/cances/gdrmomas/Ex\_qualif/Geochimie/Documents/Benchmark-MoMAS.pdf.

[] Parkhurst, D.L., Kipp, K.L., and Charlton, S.R., 2010, PHAST Version 2—A program for simulating groundwater flow, solute transport, and multicomponent geochemical reactions: U.S. Geological Survey Techniques and Methods 6–A35, 235 p.