**muFlowRT a coupling of OpenFoam and Phreeqc for groundwater flow and reactive transport.**

NB: part in red are to be done

# Installation

Windows

In the installation folder of Orti3D for windows there is a windows version : muFlowRT.exe. It is the same as the linux version, and has been compiled with BlueCFD (including phreeqcRM).

Linux

If you have a windows computer, one of the fastest and easiest solution is to install WSL on your computer (<https://docs.microsoft.com/en-us/windows/wsl/install-win10>). Choose Ubuntu for the linux version.

You need to install OpenFoam, see <https://openfoam.org/download/>. You will install openFoam8,9 or 10.

Then you need to install phreeqcRM, first read <https://water.usgs.gov/water-resources/software/PHREEQC/PhreeqcRM_ReadMe.txt>. All documentation on functions is at <https://water.usgs.gov/water-resources/software/PHREEQC/documentation/phreeqcrm/index.html> but you may not need it.

When PhreeqcRM is installed you need to copy libphreqcrm-….so (+ the symbolic link) to /usr/lib because the c++ compiler will search at this place the libraries.

Then write source/opt/openfoamXX/etc/bashrc  at the end of your ~/.bashrc file and you’ll have access to OpenFoam environnement variables (try by typing echo $WM\_PROJECT\_DIR).

Then you’ll need to compile the muFlowRT library by unzipping it in your home folder, then go into the folder and type first the following command “wmake libraries/phaseModels” (sorry the allwmake does not work). Go in solvers and do “wmake”.

In order to run one case, unzip it in the folder of your choice (normally it is in OpenFoam/run/…) and go into the folder and just type “muFlowRT”

If you work with WSL on windows, it is very easy to have the files in the windows folders so you can manage them with the tools you know. In WSL you can cd to any C: folder and run your case, the files will be produced in your windows folder.

*Specific installation to run in parallel (with MPI)*

You’ll need cmake (use aptget to get it)

Compile phreeqcrm with the option ../configure –with-mpi=yes and after the make install copy the created libphreeqcrm-3…. To /usr/lib/libPhreeqcRM (name with upper case because we will use Cmake, phreeqc convention)

Go into the muflowRT/solvers folder then:

* Create build folder and go there (mkdir build, cd build)
* Compile with cmake; by typing cmake.. and then make. It will build muFlowRT\_mpi in this folder
* To run a case, go in the folder and type
* mpirun –n 5 ~/muflowRT/solvers/build/muFlowRT\_mpi (for 5 cores)

# Theory

## Flow

### General equation

The model includes flow, multispecies transport and reaction. The reaction part is solved with PhreeqcRM, so please refer to the Phreeqc manual for more information for chemistry. Flow and transport are solved by OpenFoam. The advantages of OpenFoam are multiple : (i) a finite volume approach allowing complex geometry, (ii) a modular approach that allows to include rapidly new equations or complex behavior and (iii) several transport solver that perform fast and accurate solutions in the finite volume domain.

For single phase flow, the model solves the Darcy equation

(1)

Where *Stor* is the storage term, *Mf* the mobility, the vector of gravity flow linked to density variation and *Q* the flow source term. Here:

where *k* is the permeability (m2), the density and the *µ* viscosity of the fluid.

The storage term is equal to 10-4 (or given as stor0) for confined flow and described below for unconfined flow.

where here ***g*** is the gravity vector.

If required the equation can be solved without the temporal term in order to reach steady state (it is done to initiate the model) using the SIMPLE solver included in OpenFoam.

In the openFoam formulation for the **equations**, if fvm:: is used before the concerned term, it means that it will be solved implicitly, while if fvc:: is used it means an explicit term. The implicit approach shall be preferred but it is not always possible to use it. The above equation is written:

hEqn (

stor.fvm::ddt(h) + fvm::laplacian(-Mf,h) + fvc::div(phiG) == fvOptions(h)

);

### Unconfined flow

If the model is unconfined, the model calculates the relative saturated thickness as

(2)

where *h* is the head, *zbot* is the bottom of the cell, and *thk* its thickness, all in meters (or other length units consistent with the rest of the model), so *sw* is dimensionless. The value of *sw* is bounded by 10-4 (dry cells) and 1-1e-4 (fully saturated). The *sw* term is used as finally it can be considered as a relative saturation of the cell, except that this relative saturation is not distributed over the whole cell. In equation (1) *sw* decreases the section for flow and thus we set in case of unconfined flow, on contrary *sw* is simply set to one over the whole domain. It can be seen here that for unconfined flow this part of the equation cannot be solved implicitly, and thus *sw* is recalculated at each time step. However one must be careful as in OpenFoam all the calculations are for a volume and here the unconfined properties apply only to the horizontal flow, this is why we use the values of hydraulic conductivity at faces and during calculation for unconfined flow only the vertical faces hydraulic conductivity are modified.

Under unconfined conditions the water flux phiw is where S is the surface of the concerned face, this is correct as Mf includes the effect of sw.

In OpenFoam the tensor operation to set the equation for a given geometry are transparent: OpenFoam calculates the product of the tensor parameters with the surface vector of each cell face to reach the expression of *Mf* at the cell faces. However, since its creation Modflow (and also Modflow USG) behave differently: it is assumed that the hydraulic conductivity anisotropy is not oriented according to the horizontal direction but is parallel to the sediment layer, even if it is tilted. A specific option for the permeability field has therefore been written to reproduce this behavior. For the inter-layer faces the hydraulic conductivity is set to the value of Kv (vertical hydraulic conductivity) whether the face is horizontal or tilted. At the same time the harmonic averaging of K between cells to obtain the value at faces is done.

### Unsaturated medium

For unsaturated medium the major variable is *hp* the pressure head, and the equation is similar as Eq. 1 (Bear 1988):

(3)

Here the storage coefficient is replaced by *Ch* the capillary capacity, with (Nielsen, Th. Van Genuchten, and Biggar 1986):

(4)

Where *sw* has not the same meaning as for the saturated medium as it is here the water saturation, its maximum value and its minimum, with *Se* the effective saturation expressed as . and *m* are the Van Genuchten coefficient of the capillary curve (Genuchten 1980).

### Dual phase flow

In its present formulation the muFlowRT library includes multiphase but for two phases only. The mathematical formulation is the same as the one provided by Horgue et al. (2015), as this part of the library comes from their work, which can be expressed as follows:

(5)

And

(6)

Where is the gas pressure, the porosity, and the capillary and water fluxes across cell surface respectively. Here the fluid mobility *Mf* is the total mobility which includes water and gas.

The two equations are solved iteratively with the *P* term being implicit and the flux term being explicit in eq.5. For eq. 6, the *sw* term is implicit and the other terms are explicit.

The classical formulation of relative permeability is used with the Mualem approach (Mualem 1986).

### Immobile NAPL

## Transport

The transport of any component or species is based on the following equation (Zheng and Bennett 2002):

(7)

Where is the concentration in the fluid phase i (aqueous or gas), the fluid content ( for unconfined or for unsaturated medium), the fluid phase flux and the dispersion tensor. includes both dispersion and diffusion within the given phase. In the gas phase, the effective diffusion can be calculated with different formulation but generally we use the Millington and Quirck formulation (Millington 1959). *Q* is the source term. Here all terms are implicit, except which is explicit, coming from the resolution of the flow equation.

If *n* components are present like in reactive transport, *n* equations similar to eq.7 are written and solved separately.

For unit consistency Q is here in kg.m-3.s-1 (or mol.L-1.s-1 when concentrations are used by Phreeqc). In case of unconfined situation is replaced by .

Automated Time stepping for transport is linked to the assigned to dCmax and dCresiduals assigned in system/controlDict file.

The source term can be associated to water flow and thus linked to the flow equation source term (for concentrations in injection wells or GHB boundaries for instance).

NB: for dry cells we use the same strategy as in modflow 6, the mass reaching a dry cell is reinjected at the first wet cell below it (<https://pubs.usgs.gov/tm/06/a61/tm6a61.pdf> , p.25)

For Temperature, the equation is similar:

(8)

, and are the thermal capacity (J/kg/K) of the different media, the density (ML-3) and (MLT-3K-1) the thermal conductivity of the medium. Here the transport through the water vapor and the vaporization is neglected

This can be simplified by considering that the gas phase can be neglected in thermal exchange and that the porosity changes slowly.

## Reactions

As already stated the reactions are solved by PhreeqcRM, which is included in the muFlowRT library, but Openfoam does not see what happens in Phreeqc. This approach implies that only the operator splitting can be used, i.e. a transport step is done for all species and then Phreeqc is run and the concentrations of all species are sent back to OpenFoam. This means that **the user has to set a correct time step** for reactions to occur, there is no other option for operator splitting.

PhreeqcRM uses total component amount (i.e. the sum of all complexes of one element) as input and output. In order to solve for ionic strength and redox species, the transport also includes the amount of water, oxygen, hydrogen and charge imbalance. Therefore the total number of components is equal to the sum of all components + 4, but for instance N(5) and N(-3) account for only one component. It also means that the total number of components will not be the same as the one that can be seen in a phreeqc file. It also means that the initial concentrations of species need to be calculated from a phreeqc run.

The dissolved concentrations Cwi are in mol/L.

### Gas phase

The gas phase transfer is calculated by phreeqc and then transferred to OpenFoam. For transport calculation, the mole number of one substance relative the total number of moles can be considered as the ratio of the partial pressure divided by the total pressure and thus the concentrations in the library has been set to a proportion (equivalent to fugacity for perfect gases):

This formulation is used as it allows the transport step to be independent from the pressure equation resolution.

The number of moles in gas in cell is

Where is gas volume (L) in general per/L of medium and the molar volume. This formulation is independent of the perfect gas law as the molar volume can be obtained through the Peng Robinson equation for instance. The molar volume is calculated by Phreeqc. The moles transfer between the liquid and gas phase (Henry’s law) is also calculated by Phreeqc.

The partial pressure of each compound can change during the reaction and thus the total pressure can also change. The partial pressures resulting from phreeqc calculation are summed up to reach the total pressure and this value, transformed to Pa. In multiphase flow, the pressure is used as the initial pressure for the next flow step calculation. In unsaturated medium, the total pressure is fixed to 1 atm and supposed to stay constant due to rapid gaz exchange in the soil.

The coupling is presented in the figure



Figure 1 : coupling between Openfoam and Phreeqc, all symbols are defined in the text.

### Immobile compounds

In MT3DMS and thus PHT3D immobile compounds were used. They are used for instance for a NAPL phase that can be present in the medium, that is immobile but can dissolve in the aquifer. This concept is simpler than adding a true phase, and allows doing most of the LNAPL calculations (often the NAPL lens is immobile). This is done by using a file “constant/options/immobile” that contains the name of the species that are immobile, spelled as in the intiChem.pqi file. MuFlowRT reads that file (if present) and the associated components are not transported. (added 27/12/23)

### Coupling between equations

As shown in the previous figure the viscosity and density of the fluid are used in flow equations. This allows to modify them according to the results of transport or chemistry. This is done through a general approach, shown in Figure 2. As shown in the figure there is a flag to activate the coupling and then a function that defines the form of the coupling (this equation can be, up to now in a linear or polynomial form (not all have been tested however).

The Rhow\_Cw option state how the density of water changes with the concentration of one species, this can be used for salt intrusion for instance.

The Muw coupling can be done for two major situations: the role of the temperature on the water viscosity (classical increase of hydraulic conductivity with heat) and the injection of polymer which changes the viscosity with its concentration.

The same approaches can be done for gases. One option not presented in the Figure is the possibility of changing the gas diffusion coefficient with Temperature (De\_T)

Sw\_water is using the water to vapor change that is calculated by Phreeqc (considering the local temperature) in order to change the value of the saturation.

Finally Eps\_Cs is used to modify the value of both the porosity (eps) and premability (k) due to precipitation or dissolution of a mineral (like for instance with the Kozeny-Karman approach).



Figure 2 : representation of all functions that couple the different equations

## Implementation in OpenFoam

### Internal and boundary conditions

In OpenFoam boundary and internal conditions are differentiated (not the case in modflow series). In order to simplify the approach (for the interface also) we use only internal conditions. The boundary conditions are all set to zero flux, and at the boundaries a flux (or fixed value) of water, heat or component are used to mimic a true boundary condition. This may be an over simplification for some very specific cases but for all real cases that we’ve done, it is working correctly.

Three types of conditions for flow and transport can be set:

* A fixed value (Dirichlet type) like fixed head pressure or concentration (in dissolved or gas phase)
* A fixed rate (Neumann type): discharge or recharge Q or mass flux.
* A rate dependent of the equation variable (Cauchy type), Q = Sp.x +Su where x is the equation variable.

For a fixed value we use the setReference approach that allows to set specific values in an equation before solving. For Neuman and Cauchy, the corresponding formulation is where V is the variable (h, p, Cw or Cg, T). In OpenFoam the *Sp* and *Su* terms can directly be added to the equations (previously there was an fvOption approach to do that but it is more complex and too slow).

The muFlowRT library uses a formulation of the internal and boundary conditions similar to the Modflow one, i.e. wells, rivers, general head boundary (GHB) and drains. The three conditions are similar, with where Cond is the conductance (m2/s) and H0 is the head value fixed at the river, drain or GHB. The drain conditions is specific in the sense that the flow exists only for h>H0. The river condition is specific in the sense that the river also provides a bed value, so when h is below a given value (<H0) the exchange stops. At present (19/06/22 version) in muFlowRT GHB and drain condition are similar to Modflow ones while river is treated as GHB (should consider river bed height, not done at present). In that case and . In fact, as written previously for unit consistency the Su and Sp terms are divided by the cell volume.

# Geometry

The geometry of the problem is set by OpenFoam. The objective is to be able to construct complex unstructured geometry. OpenFoam has its own concepts that can be found in the Openfoam manuals. One important point is that when unstructured mesh is used one cell face is shared by only two cells: the option of nested mesh cannot be implemented, but true unstructured are generally better to solve transport than nested grids.

Although OpenFoam can deal with any 3D shape, in muFlowRT we restrict ourself to layers sharing the same mesh. A 2D mesh is projected down to all layers. However the position of top and bottom of each layer may vary in x,y space (z is the vertical). Openfoam does not know the concept of layers, this is only existing in muFlowRT. The layers are numbered along increasing z (oriented vertically in a positive direction).

For unstructured mesh we often use Voronoi polygons (deduced from Delaunay triangulation) but, this is done in the ORTi3D interface and in muFlowRT other geometry can be used (triangular for instance, although it may create problems in transport calculations). MuFlowRT does not provide other tools for geometry than the one present in classical Openfoam (for instance you can use blockMesh to generate your grid).

In order to deal with cross sections and radial models, the considered geometry in muFlow is horizontal but the gravity vector is modified (gravity in the y direction). Therefore these models include only one layer.

# Structure of the library

MuFlowRT has been developed in parallel with enhancement of the ORTI3D (orti3d.ensegid.fr) interface, and thus all files given below can be produced directly from the interface. We advise the user not to try to produce unstructured mesh by his own, it is very time consuming. ORTI3D is restricted to regular mesh and unstructured mesh with Voronoi polygons.

The structure of the library is given below. The only .C file is muFlowRT.C file, it uses all files below that .H files (this structure with a major C file and a lot of H files is used in general in OpenFoam). The files below are in the solvers folder

|  |  |
| --- | --- |
| createFields | Create all fields needed by the equations |
| createThetaFields | fields specific to unsaturated medium |
| create2PhaseFields | fields specific to dual phase |
| updateThetaFields | update the theta fields for each time step |
| update2PhaseFields | update the dual phase fields for each time step |
| hEqn | the equation for head |
| hstEqn | the head equation for steady state |
| hpEqn | the head equation for unsaturated medium |
| pSEqn | double pressure and saturation equation (Impes) for dual phase |
| flow | this file allows to choose among the different flow options |
| CoatsNo | calculate the coats and courant number for dual phase (to obtain the time steps) |
| readPicardControls | When Picard option is used (for unsaturated case) |
| myFunc | Generic option for calculating the function for coupling between equations |
| readFunc | file to read the functions |

One can also find two exploratory files : nnBase and chem\_nn that are used for neural network approach.

In the solvers/transport the following .H files can be found

|  |  |
| --- | --- |
| createCFields | Create the fields required for transport of a tracer |
| createTFields | Create the fields required for temperature |
| createCwiFields | Create the fields required for multi dissolved components transport (using Phreeqc) |
| createCgiFields | Create the fields required for multi gas transport (using Phreeqc) |
| Ceqn | Transport of a tracer equation |
| Teqn | temperature/heat equation solver |
| CwiEqn | Transport on n components |
| CgiEqn | Transport of n gases |
| setDeltaTTrsp | used to modify the time step used for transport |

For chemistry the /phreeqc/initPhreeqc file is the only one present, it provides a class (freak) that allows the intialization and run of PhreeqcRM.

# Files/Fields/Variables

## Openfoam fields

In Openfoam the variables are fields and depending on their creation they are used internally and printed at output time steps, or remain only internal

*First we list the fields that will be written to the disk.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Basic files/variables for Openfoam** | | | *Unless specified all are volScalarField* | |
| folder | file | comment | |
| 0 | Uw | required even if not used | |
|  | Ug | required even if not used | |
| constant | transportProperties | model parameters | |
|  | g | gravity vector | |
|  | eps | porosity | |
| constant/ polyMesh | points | xyz coordinates of the points | |
|  | faces | points contained in each face | |
|  | owner | cell to which the face belong (ordered as faces) | |
|  | neighbour | neighbour cell of the face | |
|  | boundary | faces number that belong to a boundary | |
| system | fvSchemes | schemes for the solver (linear,vanLeer...) and interpolation (upwind..) | |
|  | fvSolutions | matrix solvers (PCG, BiCGStab…) | |
|  | controlDict | input output format, time.. | |
|  | blockMeshDict | describes mesh if simple mesh built through opf | |

The following fields are used for flow problems

|  |  |  |
| --- | --- | --- |
| **Files/fields for a pure flow problem** | | |
| 0 | h | head (m) |
|  | p | pressure (Pa) |
|  | sw | for unconfined : it will be the % of height occupied by water, for unsaturated and dual phase : the water saturation |
|  | hp | this is the pressure head (m) for unsaturated problem |
| constant |  |  |
|  | Kh | Horizontal permeability (m2) |
|  | Kv | Vertical permeability (m2) |
|  |  |  |
| **for a 3D problem (required for an unconfined problem)** | | |
|  | thk | thickness of each cell (m) |
|  | zbot | bottom of the cell (m) |
|  |  |  |
| **for an unsaturated flow problem (in createThetaFields)** | | |
|  | sw\_min | minimum water saturation |
|  | n\_vg | value of the n parameter of the van genuchten function |
|  | alpha\_vg | value of the alpha parameter of the van genuchten function |

For transport /temperature the following files are present

|  |  |  |
| --- | --- | --- |
| **for one substance transport (in transport/createCfields)** | | |
|  | C | concentraiton of the substance (kg/m3) |
|  | rhos | solid density (kg/m3) |
|  | Kds | Distribution coefficient vs solid (m"/kg) |
|  |  |  |
| **for thermal (in transport/createTfields)** | | |
|  | T | Temperature |
|  | cps | heat capacity of the solid [L2 T-2 °C-1] |
|  | lbdaTs | Thermal conductivity of the solid [M L T-3 °C-1] |
|  |  |  |
| **In presence of reactions (in transport)** | | |
|  | Cw[i] | concentrations of components in water (mol/L) |
|  | Cg[i] | concentrations of components in gas (% mole sof gas phase) |

Here are the fields or scalar that will be created from the TransportProperties file (or by simple default values during createFields)

|  |  |
| --- | --- |
| **basic parameters (Transport properties)** | |
| phase.w/g | Provide density (rhow, rhog) and viscosity (muw,mug) of the corresponding phase |
| ncell\_lay | number of cells per layers |
| nlay | number of layers (openfoam does not know it) |
| flowStartSteady | 1 is the flow has to be equilibrated before the start of the run |
| flowType | 0: non flow, 1: confined, 2: unconfined, 3 : unsaturated, 4: dual phase |
| stor0 | storage coefficient for confined medium |
| *flags* |  |
| activateTransport | 1 if transport is activated |
| activateCapillarity | 1 if capilarity is activated |
| activateThermal | for T calculation |
| activateReaction | 1 to activate the reaction (phreeqqc) |
| reactionSteps | this is number transport time steps before doing reaction |
| *for transport* |  |
| alphaL | longitudinal dispersivity |
| alphaT | transverse dispersivity (horiz and vertical) |
| Dw0 | diffusion coefficient in water (m2/s) |
| Dg0 | diffusion coefficient in gaz (m2/s) |
| lbdaw | decay constant in water |
| *for temperature* |  |
| cpw | scalar: heat capacity of the water |
| lbdaTw | scalar:heat conductivity of water |

The following variables are internal, and used for intermediate calculations (they are the importna tones, other may exist but are not detailed here)

|  |  |
| --- | --- |
| **Internal fields** | |
| Kf | this the value of the permeability at the faces |
| unity, a, b | dummy fields |
| M0f, Mf | mobility (k.rho.g/mu) at faces |
| stor | storage coefficient (1/m) used in both the saturated the unsaturated equation |
|  |  |
| **for unsaturated and dual phase** | |
| thetaw | water content |
| thetag | gas content |
| m\_vg | 1-1/n\_vg |
| sw\_max | maximum saturation( fixed at 1-10^-6 now) |
| Se | effective saturation ((sw-swmin/(swmax-swmin)) |
| krw | relative permeability to water calculated from K and Se (-) |
| krwf | same as above but on surfaces |
| dkrwdS | derivative of krw vs Se |
| Lf | gravity part (k.rho/mu) of equation |
| phiGr | the gravity flux |
|  |  |
| **specific to dual phase** | |
| krg | relative permeability to gas calculated from K and Se |
| krgf | same as above but on surfaces |
| pc0 | capillary entry pressure (Pa) calculated from alpha\_VG |
| dpcdS | derivative of pc vs S |
| dkrgdS | derivative of krg vs S |
| Mwf | Mobility of water on cell surfaces |
| Lwf | same for Lf |
| phi, phig, phiw | total, gas water flux per face |
| phiPc | capillary flux |
| gradpc | gradient of capillary pressure |
| Deffw | effective diffusion coefficient in water (m2/s) |
| R | retardation factor |
| **specific to transport/thermal** | |
| Deffg | effective diffusion coefficient in gas in porous medium (m2/s) |
| lbdaT | sediment thermal conductivity (use w and s) |

Some files are specific to chemistry

**Units**. In openFoam the units are set in the international system, with the convention kg, m, s for the first three unit that we use, *i.e.* [1 -1 -2 0 0 0 0] will represent *kg.m-1.s-2*. This also means the results will be given for a number of second since the start.

## Chemistry specific files

**Ractiv**: this file contains a list of the cells number where the chemistry will be calculated

The main folder includes the files required for phreeqc:

**phreeqc.dat** the phreeqc database

**initChem.pqi** : this is the phreeqc file of your problem, formatted as in phreeqc (it can be used in phreeqc for tests)

**phqinit.txt** and **phqfoam.txt** files are used to specify to Openfoam the dimensions of the chemical problem and to define the starting composition of the cells for phreeqc. Phqinit contains only a number cell equal to the number of solutions in your problem. It is used to make a first run of phreeqc that is used to initiate the solution composition in OpenFoam. The files are structured as follows

1st line

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nb of cells (or only the active ones, if a sub domain for chemistry is defined) | Nb of dissolved components | Nb of gas components | Nb of different solutions | Units for phase exchange.. |

Then they are 7 lines that contain the number for:

|  |  |  |
| --- | --- | --- |
| Group (not written) | 1st number | List of numbers |
| Solution | 0 or -1 | One number or a list of ncells values |
| Equilibrium phases | “ | “ |
| Exchange | “ | “ |
| Surface | “ | “ |
| Gas phase | “ | “ |
| Solid solutions | “ | “ |
| Kinetics | “ | “ |

For each line, whether the first number in the line is 0 and then all cells have the same solution number (use -1 if the group is not used), or the first number is -1 and it must be followed by the solution number for each cell.

**Solutions**: these files are written automatically by muFlowRT: it contains the chemistry of the solutions present in the problem. (needs to be written by muFlowRt after the first phreeqc run because we need 2O, H, O and e- amounts).

**Gases**: same as above but for gas components concentrations

The chemistry associated variables are given below. These variables are generally vector of floats to be exchanged with phreeqc (there is always a transfer from Openfoam fields to c vectors for phreeqc, we did not find another solution)

|  |  |
| --- | --- |
| **specific to chemistry** | |
| *normally all variable have ractive size (not ncell)* | |
| nxyz | number of cells (=ncell[ractive]) |
| ph\_ncom,ph\_gcomp | nb of dissolved and gas compounds |
| ph\_nsolu | nb of solutions |
| gvol | gas volume (eps.(1-sw)) |
| atmPa | a scalar for converting atm to Pa |
| c\_ph | dissolved concentrations in phreeqc format |
| gm\_ph | nb of moles in gaz, phreeqc format |
| rchange | index for places where change chem -> calc chemsitry |
|  |  |
| *in the phreeqc class, accessed by muFlowRT.c* | |
| freak.c | dissolved concentration |
| freak.g | gas comonent partial pressures |
| freak.gm | moles of component in gas |
| freak.spc | species in selected output |

## Boundary and internal conditions

The values of the parameters necessary for the internal conditions for each cell and time are contained in options files that have the same name as the cellSet. For flow the names are hwel, hrch, hriv, hdrn and hghb for the five types of conditions (Wells, recharge, rivers, drain and general head respectively). Indual phase the names start with p (only pfix and pwel exist). For transport the names are the same but starting with C, temperature T. For chemistry there are two types of files: starting with S, is for solution number and G the gas assemblage number. In fact these files provide the solution number and muFlowRT uses the solution file (cited above) to retrieve component concentrations from solution number and solution composition. The files are ordered as follows

In the first line there is the number of layers and the number of cells per layer, then for the next lines the format is “Time (d) cell\_nb Su Sp “ # one line for each cell in each time.

***Time units here are in days***. This is done because seconds are difficult to read. All data are transformed to seconds internally.

Change since 23/09 : the format of the files are now binary (float single precision), this is because when the file were too large the memory was filled. Therefore they need to be written by an application (ORTI3D does the job for us, but python can also do it easily and the format can be transferred from windows to linux). In order for the user to be able to verify, all files are doubled, there is an \*\_a equivalent for the ascii format: these are not used by muFlowRT.

Example:

0 0 0 0

1 43 -3e-4 0 // this is for one day and first cell of the cellSet (cell nb 43)

1 57 -4e-4 0 // this is for one day and 2nd cell of the cellSet (cell nb 57)

….

17820 -2e-4 0 // this is for time=two days and first cell of the cellSet

…

Last time+0.01 0 0 0

Where Su (here) is the injection rate in m3/d and Sp is the term that allows to vary the discharge/recharge in proportion to the head value (Cauchy type of condition). For recharge the rate is the daily recharge thickness multiplied by the cell area.

There can be a combination of two input files for some cases. For instance, for an injection well the injected discharge is in the qwell file, while the composition of the injected solution is in Swell (gas in Gwell). **It is the user responsibility** to set both values at the same place (it is difficult to verify this in the interface). For pumping wells, muflowRT calculate internally the amount of water removed and multiply it by the local concentration to get the outgoing mass flux. This is done implicitly (using the Sp term) so normally there should not be problem at wells. It is the same for ghb boundary conditions.

For recharge it is the same if flow recharge is specified in some zone, the tracer concentration or the solution chemistry of recharge **must** be provided for the same zone. If there is a background recharge for flow, the background for chemistry will be taken.

In pwel the variable is the gas flow rate in the well. All fix are in the same units as the variable (p in Pa, h in m, C in kg/m3, S and G have no units).

The time can be different for each file. For instance in hfix there can be only the time 0 and the final time (+0.01 day), while in the recharge file the value can change every day. This allow a high flexibility.

The Constant/polymesh/sets folder contains several cellSets that are associated with one process, as described above. For instance hwell is the cellSet that contains the list of all cells that contain wells. The list of cells must correspond to the one in the file with the same name in constant/options.

## System

### fvSchemes

In openFoam each part of an equation has a name and thus a scheme can be specified for each part of any equation. This is valid for the temporal scheme, the divergence and laplacian terms. The temporal scheme allows for instance to choose between Euler or CrankNicolson temporal schemes. For instance below a vanLeer scheme is declared for dissolved concentration. The file also allows to set the way the interpolation of spatial variable is done at the cell faces (below upwind interpolation for kr).

ddtSchemes { default Euler; ddt(sw) CrankNicolson 0.5; }

gradSchemes {default Gauss linear;}

divSchemes {default none;div(phiw,Cw) Gauss vanLeer;}

laplacianSchemes {default Gauss linear corrected;}

interpolationSchemes { default linear; krg upwind phig; krw upwind phiw; }

### fvSolutions

this file allows to select the matrix solver for each equation, for instance

h {solver PBiCGStab;preconditioner DIC;tolerance 1e-12;relTol 0;}

where we can see the choice of the solver, preconditioner tolerance…

If picard iterations are used the parameters are set here.

### ControlDict

The controlDict allows to specify the time steps, writing steps, start and end time and the way to write the output files. The time steps constraints can be specified as a simple max time steps or is also used to set the constraints in terms of courant number (or residual concentrations variation between two time steps).

Interestingly the controlDict parameters can be changed during the simulation (usefull for a restart or increase time steps).

Using variable size time steps is tricky, it requires several files (implemented in ORTI3D)

### blockMeshDict

this file is used to build the geometry for simple geometries, see the openfoam documentation or examples to see how it works. blockMeshDict is used by the program blockMesh to produce the files that are in constant/polyMesh (see below).

# For developers

There is a curious but useful use of .H files in a lot of OpenFoam libraries. In fact the .H contain most of the calculations while normally these are in the .C. This allows to simply include them when a calculation is required. We mostly use this approach.

## Fields

There are scalar, vector and tensor fields, they can be defined by cells: e.g. volScalaraField but also on the faces surfaceScalarField. During the definition the field can be defined as fieldName.timeConstant, they will remain constant and be in the constant folder, or fieldName.timeName () they can change over time and be stored in the time folders. During the creation of a field one can set its dimension an its initial value. If no initial value is specified openfoam will ask for the file.

In the library, K is a tensor (to account for anisotropy), so the mobility is also a tensor. Then in order to have for instance the mobility multiplied by gravity, the operation is “Mf & g” which is a tensor and not “Mf \* g” which would be valid if Mf would be a simple scalar.

## Time stepping and nonlinear equations

We did not develop a very complex approach for non-linear equations. For most of the equations we tried, as much as possible to use the implicit formulation. However, unsaturated and dual phase problems cannot be treated like this due to high non linearity. So the approach is used to use simple time step adaptation. The time stepping can be defined by the user but they are also set to respect courant number for flow (or Coats number for dual phase if this option is chosen in controldict). Fro unsaturated flow the Picard approach (inspired by Horgue et al 2015) seems to provide rapid and correct results. For transport we found that the dCmax approach was the easiest and the best. The time step is calculated in order that dCmax: the relative variation of concentration remain lower than a value set by the user. Surprisingly a value of 0.05 provides good results. This is also done for the chemistry step.

The Newton Raphson approach is not used now but could be in a future.

## Conditions

Boundary and internal conditions are a large part of a real problem and take time to solve correctly. We use the Su and Sp approach of OpenFoam (fvOptions were slower and more complex). At each internal time step the conditions are analyzed and included in the equations as required by the user.

**Be aware**: The Su term is explicit: this term is added to the matrix as a flux (the right side in the classical formulation), while the Sp term is implicit, it creates a proportionality between the actual term in the matrix (left hand) and the Q term (right side).

For most of the cases we use a zeroGradient boundary conditions at the domain boundary, which allow consistency between cells inside the domain and cells at boundary (it is possible to put a well in cell at a boundary.

## Phreeqc coupling

The phreeqc coupling is done through the initPhreeqc.H file but it is a little bit tricky, as the initChem.pqi file needs to be read in order to set the number of components in the simulation.

The operations are done in the following order:

* muflowRT reads the files constant/options/ractiv (active cells), the phqfoam.txt file (distribution of initial solutions exchange…) that is internally called ph\_data. The first line of ph\_data is used to specify the nxyz, number of cells in the model and n\_comp, the total number of components
* then ph\_data are sent to freak class (setData), the database ifsset (setDB) using the address of phreeqc.dat file and the pqi file (setChemistry) using the initChem.pqi file.
* Freak.init() is run once for a number of cells=number of solutions to calculate the composition of the initial solutions
* muflowRT uses the initial solutions to write constant/options/solutions file that contains the components of each solution (written sequentially by solution number), this will be used for boundary/internal conditions. When gas are present an additional file gases is written for initial gaseous components.
* freak.init() is run again but for the whole domain cells. Phreeqc stores the solid phases components (minerals, exchangers, surfaces)
* createCwiFields.H is run to build the n\_comp variables Cwi in OpenFoam (the Cwi variables have the size of the total domain, not ractiv length)
* run the flow equation
* run the transport of n\_comp species (CwiEqn)
* finds the place where chemistry has changed (any species relative changing of more 5e-5) stored in rchange array
* transfer the Cwi concentrations to a one dimensional array c\_ph (n components one after the other with ractiv size for each component) and provides if to freak class through the setC function
* specify where to calculate through freak.setCalc which uses setSaturation: 1 for calculating the chemistry and 0 for not calculating (it seems that there is no other way to inactivate calculation in phreeqcRM)
* set time step and runs phreeqc
* reset all Cwi concentrations to 0, and then get the freak.C concentrations and copy them to Cwi. The reset to 0 is required because transport will calculate concentrations around the ractiv zone and we don’t want to lose time with these concentrations

PhreeqcRM does not want null conc at the beginning, put 1e-12

For **gas exchange** : be carefull : gasphase Units are fct of RV, set gvol and wsat before attributing C or gm. It is required to set both wsat and gvol.

Seelcted\_Output variables are exported to a file called Species, be carefull the order of the species can be changed (seems that totals appear at first)

## Variable coupling

In natural system water or medium properties can vary with other variables, like viscosity that varies with temperature. This is done through one file called coupling, located in constant/options, which has the following structure (the first line is absent in the true file):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Y | X | Xref | type | Nb of parameters | a0 | a1 | a2 | a3 |
| muw | T | 25 | linear | 1 | -1e-3 |  |  |  |
|  |  |  |  |  |  |  |  |  |

For instance in the first line muw varies linearly with temperature, with a coefficient = 1 when T = Tref (here 25°C), and a slope of 1e-3, i.e. at 35°C the multiplying coefficient will be 1-1e-3\*10=0.99. or fact= 1+a0.(T-Tref)

This file is read by the muFlowRT.C that stores all couples. Then the different coupling are used in different equations. For instance the variation of muw with temperature (or concentration) is used in the flow equation (generally h), this is why this part is implemented in the file plugin\_H.H. This plugin look at the different couples that concern the h equation (muw rhow, K) and modifies the values of these Y fields in relation with the X field, to finally calculate a global factor that modifies the mobility (Mf).

For permeability variation with mineral precipitation or dissolution, there is a need to create a specific field that accumulates the mineral phases precipitated or dissolved along time. Indeed the factor (fact) that will multiply Mf is calculated at each time step using state variables like T or C, so we created a field called deltaMVol that stores the volume of mineral precipitated or dissolved since the beginning of the simulation. This field is present in the plugin\_H.

# Annex

Phreeqc

If you ever happen to want to link against installed libraries in a given directory, LIBDIR, you must either use libtool, and specify the full pathname of the library, or use the '-LLIBDIR' flag during linking and do at least one of the following:

- add LIBDIR to the 'LD\_LIBRARY\_PATH' environment variable during execution

- add LIBDIR to the 'LD\_RUN\_PATH' environment variable during linking

- use the '-Wl,-rpath -Wl,LIBDIR' linker flag

- have your system administrator add LIBDIR to '/etc/ld.so.conf'

For fluxes calculation

phiw = hEqn.flux() provides correct fluxes (null balance in a cell),

while - ((Mf & fvc::interpolate(fvc::grad(h)) ) & mesh.Sf() ) does not (why, I don’t know)

Interpolation schemes possibles

CoBlended

Gamma

MUSCL

Minmod

OSPRE

QUICK

SFCD

SuperBee

UMIST

biLinearFit

blended

cellCoBlended

clippedLinear

cubic

cubicUpwindFit

downwind

filteredLinear

filteredLinear2

filteredLinear3

fixedBlended

limitWith

limitedCubic

limitedLinear

limiterBlended

linear

linearFit

linearPureUpwindFit

linearUpwind

localBlended

localMax

localMin

midPoint

outletStabilised

pointLinear

quadraticFit

quadraticLinearFit

quadraticLinearUpwindFit

quadraticUpwindFit

reverseLinear

skewCorrected

upwind

vanAlbada

vanLeer

weighted

)

PhreeqcRM surface

PHREEQC version 2 and 3 have the same units convention for SURFACE

The units in a SURFACE definition result in a number of moles of surface sites. Moles are either defined directly (default) as a product of sites per nanometer^2 times surface area (with an internal factor to convert sites to moles and nanometers to meter^2), or as a proportion of a mineral or kinetic reactant. Regardless of the different input options, the result is moles of sites, not moles per liter water or moles per representative volume, just moles.

The following is excerpted from the documentation for the PhreeqcRM method SetUnitsSurface:  
  
**In PHREEQC input, surfaces are defined by moles of surface sites (Mp). SetUnitsSurface specifies how the number of moles of surface sites in a reaction cell (Mc) is calculated from the input value (Mp).  
Options are 0, Mp is mol/L of RV (default), Mc = Mp\*RV, where RV is the representative volume (SetRepresentativeVolume); 1, Mp is mol/L of water in the RV, Mc = Mp\*P\*RV, where P is porosity (SetPorosity); or 2, Mp is mol/L of rock in the RV, Mc = Mp\*(1-P)\*RV.**

The SetUnitsSurface option is used when the PhreeqcRM method InitialPhreeqc2Module or InitialPhreeqcCell2Module is invoked. The moles in a PhreeqcRM cell are calculated from the PHREEQC SURFACE definition using the specified units option.

As Pht3d is in mol/L bulk volume, the only solution is to set option to 1 in phreqcRM (initChem.pqi) and transform the amounts from Pht3d to mol/L (considering porosity)