Final Project Report

IORSim – IOR chemistry simulator coupled to commercial reservoir simulator

Project number and location (UiS, NORCE, IFE): O-02001-01 (IFE)
Project duration: 2013-2021
Project manager: Jan Nossen (IFE), Aksel Hiorth (NORCE/UiS)
PhD students and postdocs: Felix Feldmann
Other key personnel: Jan Sagen, Jan Ludvig Vinningland, Jan Nossen, Jarle Haukås, Terje Sira, Børre Antonsen, Steinar Groland, Egil Brendsdal

1. Executive summary
2. Introduction and background
   (Project initiation, tasks involved, collaboration (national and international), user partner involvement if relevant)
3. Results
   (Description and discussion, research-/user partner involvement)
   Description of most important sections of the program

Contents

Executive Summary

Commercial reservoir simulators cannot simulate the complex chemical interactions that we know from lab experiments take place and are important to model to predict the effect of EOR fluids at field scale. On the other hand, research codes cannot simulate a realistic field with a complicated production history. IORSim tries to bridge the gap between state-of-the-art research codes with a detailed description of the chemistry and commercial reservoir simulators that can simulate a field with a long and complicated production history. IORSim can be run in two modes (i) forward mode (ii) backward mode. In the forward mode IORSim advects species passively along the flow lines predicted by the reservoir simulator. In this mode one can predict reservoir pH and scale potential in the reservoir and in production wells. In the backward mode IORSim is run in parallel with the reservoir simulator, IORSim calculates the change in water chemistry and updates the flow functions used by the reservoir simulator.

The backward mode has been used to study the impact of low salinity water, where the change in relative permeability is based on the amount of ion exchange in the reservoir. We have also used IORSim to simulate the silicate injection (Stavland, Jonsbråten et al. 2011, Skrettingland, Giske et al. 2012, Skrettingland, Dale et al. 2014) performed by Statoil on the Snorre field, and history matched the production profile. These results are not included here as we are still discussing these results with the Snorre license. In this report we include a generic case, where we inject silicate to block several thief zones in an artificial reservoir as proof of concept.

The block sorting numerical algorithm presented here is novel, similar ideas have been presented before (Natvig and Lie 2008), but this is the first time they are used to communicate with a

commercial reservoir simulator. We have also done several tests to check that the numerical algorithm converges to the correct solution.

Introduction

IORSim has been developed as a plug-in tool for reservoir flow software to implement and visualize complex reactive transport simulations. Besides the already implemented tracer, sodium silicate, and geochemical modules, the software design supports an uncomplicated definition and implementation of new chemical modules. IORSim uses a sequential iterative approach, in which a reservoir flow simulator is used to calculate the fluid flow while IORSim calculates the non-linear geochemical reactions.

Depending on the users' requirements, IORSim can be run in a forward or backward simulation mode. During a forward mode simulation, IORSim is used as a post-processor tool which advects the chemical species transport along the flow lines. While the fluid flow of the host simulator remains unaffected, the high-speed forward mode is helpful to estimate the chemical species distribution in complex reservoir structures.

When selecting the backward mode, the host simulator and IORSim communicate at each time step to update the fluid flow depending on the chemical reactions. At the current stage, IORSim uses Eclipse 100 to implement reactive multiphase flow simulation in reservoir structures. The backward mode communication process between IORSim and the host simulator is depicted in Figure 1. The user initially creates a standard Eclipse data file (root.data), including all common Eclipse subfiles such as PVT, relative permeability, and well schedule files. After starting the simulation, Eclipses computes the first timestep and writes the output files. IORSim then pauses the Eclipse simulation to initiate the geochemical species transport and reaction computations.

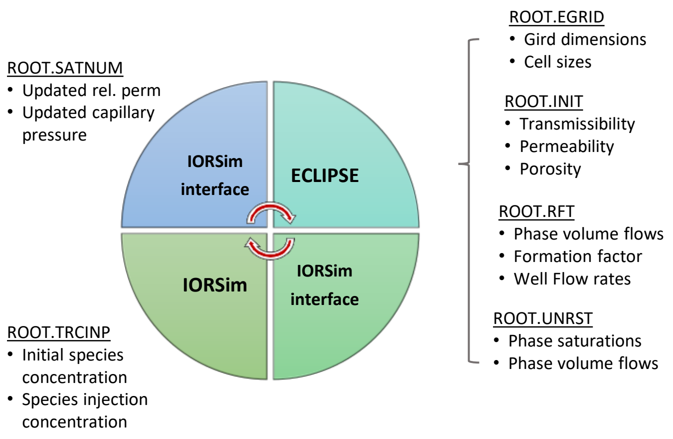


Figure 1**:** Communication process between Eclipse and IORSim

In the next sections we describe the geochemical model in IORSim, and the silicate model. Then we present some simulation results using these models. Next, we present in detail how IORSim communicate with a reservoir simulator (ECLIPSE), and how we can calculate information about passive and active tracers in the reservoir, and how we can use this approach to calculate rock fluid interactions and feed this information back to the reservoir simulator. Finally, we present the graphical user interface to IORSim.

The geochemical model in IORSim

Introduction

The geochemical model in IORSim was developed as there was a need to model rock fluid interactions from the point of injection to production. Rock fluid interactions could be one or all of the following: (i) dissolution/precipitation that could change porosity, permeability and/or wettability, (ii) ion exchange to model fluid-clay interactions (iii) surface complexation to model changes in surface potential and streaming potential in sandstones and/or chalk. The latter interactions most likely can explain water weakening effects observed in chalk reservoirs, see (Minde and Hiorth 2020). The geochemical model calculates the thermodynamic properties of the aqueous species using the same approach as (Johnson, Oelkers et al. 1992), which is based on the work by (Helgeson and Kirkham 1974, Helgeson and Kirkham 1974, Helgeson, Kirkham et al. 1981, Aagaard and Helgeson 1982). This approach allows calculation of thermodynamic properties from 0 to 1000C and 1 to 5000 bar. Although the calculations can be done at a large range of temperatures there is a limit on the salinities, typically this approach is more uncertain when the salinity is more than twice the seawater. However, the applications we are most interested in are to describe the chemical reactions that happens when seawater or low salinity water is injected into the reservoir and then this approach should be well inside the limits of the theory. The other strength with the approach used by Helgeson, is that it is quite easy to extend to ion exchange and surface complex interactions. Surface complex interactions are needed to model smart water effects (Hiorth, Cathles et al. 2010), and surface chemistry effects in sandstones (Revil, Pezard et al. 1999, Revil, Schwaeger et al. 1999).

Theory

In this section we will explain some basic theory behind the approach suggested by (Johnson, Oelkers et al. 1992). One important point is that the aqueous chemistry in a brine is very different from fresh water (or low salinity brines). The physical picture is that ions in a brine are always surrounded by other ions, thus the thermodynamical activity is different from the *ion concentration.* To determine the thermodynamic activity, one needs to solve a set of mathematical equations. The thermodynamic activity is typically a fraction of the total ion concentration and is needed in order to predict solubility of minerals, pH, ion exchange etc.

In Table 1 the result of a typical geochemical calculation is shown. We have specified the total concentration of the ions (Na, Mg, Ca, K, SO_4, Cl, HCO_3), pressure and temperature, and then the concentration of the major dissolved species can be predicted. If we look at e.g., the total concentration of SO4 we observe that 52% exists as the SO_4^{2-} component, 3% as $CaSO_4^0$, 26% as $MgSO_4^0$, 19% as $NaSO_4^-$, and 0.6% as KSO_4^-. We normally use the term “basis specie” to mean the free ion and secondary specie (or complexes) for the association between an anion and cation (e.g., $MgSO_4^0$, $NaSO_4^-$). Furthermore, it is the concentration of the basis specie SO_4^{2-} that is important for predicting solubility of minerals e.g., formation of anhydrite ($CaSO_4$ mineral), and not the total concentration. Thus, if the concentration of Na was increased or decreased it would affect the solubility of anhydrite. Another important example is the solubility of gold (Au), gold makes a complex with Cl (AuCl) and therefore gold is much more soluble in waters containing chlorine. Understanding how the water composition affects solubility and rock fluid interactions in general is therefore of great importance with respect to ore deposits and a great deal of efforts has been made to measure and model the ion-ion reactions.

Table 1: Distribution of major dissolved species in sea water, the ion pairs are usually termed complexes. The calculation is done at 130C and a pressure of 8 bars

Ion	Molality (Total)	Free Ion (percent)	Me-SO_4 Pair (percent)	Me-HCO_3 Pair (percent)	Me-CO_3 Pair (percent)
Na^+	0.48	99	1	0.04	0.006
Mg^{2+}	0.054	86	13	0.6	0.4
Ca^{2+}	0.010	90	9	0.6	0.1
K^+	0.010	98	12	-	-

Ion	Molality (Total)	Free Ion (percent)	Ca-anion Pair (percent)	Mg-anion Pair (percent)	Na-anion Pair (percent)	K-anion Pair (percent)
SO_4^{2-}	0.028	52	3	26	19	0.6
HCO_3^-	0.0024	76	3	14	8	-
CO_3^{2-}	0.00027	9	7	72	12	-
Cl^-	0.56	100	-	-	-	-

When simulating geochemical reactions in transport codes, like in IORSim, the traditional way of doing it is to transport only the total concentration of the chemical species. As an example, in the case of SO_4 we only transport one component, and not SO_4^{2-}, $CaSO_4^0$, $MgSO_4^0$, $NaSO_4^-$, KSO_4^- etc. To obtain the (thermodynamic) activity of sulphate or any other ion in solution, we assume that the solution is in equilibrium with all the complexes in solution and perform a calculation. The result of this calculation is like the one shown in Table 1. This approach is similar to most geochemical software packages, and also what is done in PHREEQC (Parkhurst and Appelo 1999).

Note that even if we obtain the activity of the basis species from equilibrium calculations, the interactions between the rock, oil or a gas phase is not in equilibrium. It is only the interactions that takes place in the water phase that is in equilibrium.

Mathematical formulation

IORSim calculates the new total concentration in a block from equation (33), then the geochemical solver determines the thermodynamic activities by solving a set of equations. To make these calculations more understandable we will specify to a specific chemical system. In the following the total concentration c_j is calculated from the first term in equation (35), thus the total concentration depends on the time step and flux.

Calcite and dolomite in distilled water

In order to describe the geochemistry when calcite ($CaCO_3$) and dolomite ($MgCa(CO_3)_2$) equilibrates with distilled water, we choose a basis set (this set is not unique): H^+, Ca^{2+}, Mg^{2+}, HCO_3^-, H_2O. The activity of H_2O is usually set to a specific value (we will put it equal to 1 in the following). The secondary species can then be expressed as a linear combination of this basis set:

$$
\begin{array}{llllllll}
\mathrm{Ca\,HCO_3^+} & \rightleftharpoons & \mathrm{Ca^{2+}} & + & \mathrm{HCO_3^-}, & & & \\
\mathrm{Ca\,OH^+} & \rightleftharpoons & \mathrm{Ca^{2+}} & + & \mathrm{H_2O} & - & \mathrm{H^+}, \\
\mathrm{H_2CO_3^0} & \rightleftharpoons & \mathrm{HCO_3^-} & + & \mathrm{H^+}, & & \\
\mathrm{CO_3^{2-}} & \rightleftharpoons & \mathrm{HCO_3^-} & - & \mathrm{H^+}, & & \\
\mathrm{OH^-} & \rightleftharpoons & \mathrm{H_2O} & - & \mathrm{H^+}, & & \\
\mathrm{Ca\,CO_3^0} & \rightleftharpoons & \mathrm{Ca^{2+}} & + & \mathrm{HCO_3^-} & - & \mathrm{H^+}, \\
\mathrm{Mg\,HCO_3^+} & \rightleftharpoons & \mathrm{Mg^{2+}} & + & \mathrm{HCO_3^-}, & & \\
\mathrm{Mg\,CO_3^0} & \rightleftharpoons & \mathrm{Mg^{2+}} & + & \mathrm{HCO_3^-} & - & \mathrm{H^+}, \\
\mathrm{Mg\,OH^+} & \rightleftharpoons & \mathrm{Mg^{2+}} & + & \mathrm{H_2O} & - & \mathrm{H^+}.
\end{array}
\tag{1}
$$

For each of the reactions above there is a corresponding law of mass actions with a known dissociation constant. Using the law of mass action we can write down mathematical equations for all the reactions in equation (1):

$$
\log_{10}\begin{bmatrix} K_{\mathrm{Ca\,HCO_3^+}} \\ K_{\mathrm{Ca\,OH^+}} \\ K_{\mathrm{OH^-}} \\ K_{\mathrm{H_2CO_3^0}} \\ K_{\mathrm{CO_3^{2-}}} \\ K_{\mathrm{Ca\,CO_3^0}} \\ K_{\mathrm{Mg\,HCO_3^+}} \\ K_{\mathrm{Mg\,CO_3^0}} \\ K_{\mathrm{Mg\,OH^+}} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ -1 & 0 & 1 & 0 & 1 \\ -1 & 0 & 0 & 1 & 1 \end{bmatrix} \cdot \log_{10}\begin{bmatrix} a_{\mathrm{H^+}} \\ a_{\mathrm{Ca^{2+}}} \\ a_{\mathrm{HCO_3^-}} \\ a_{\mathrm{H_2O}} \\ a_{\mathrm{Mg^{2+}}} \end{bmatrix} \tag{2}
$$

$$
-\log_{10}\begin{bmatrix} a_{\mathrm{Ca\,HCO_3^+}} \\ a_{\mathrm{Ca\,OH^+}} \\ a_{\mathrm{OH^-}} \\ a_{\mathrm{H_2CO_3^0}} \\ a_{\mathrm{CO_3^{2-}}} \\ a_{\mathrm{Ca\,CO_3^0}} \\ a_{\mathrm{Mg\,HCO_3^+}} \\ a_{\mathrm{Mg\,CO_3^0}} \\ a_{\mathrm{Mg\,OH^+}} \end{bmatrix},
$$

$$
\log_{10}\mathbf{K} = \mu\cdot\log_{10}\mathbf{m} - \log_{10}\mathbf{n},
$$

$$
n_i = 10^{\sum_{j=1}^{N_b}\mu_{ij}\log_{10}m_j - \log_{10}K_i},
$$

Note that we use the symbol m_i for the basis species, and n_i for the secondary species, the activities are related to the concentrations in the following way

$$
\begin{aligned}
a_i &= m_i\gamma_i\,,\ \text{where the activity coefficient, } \gamma_i \text{ is defined as:} \\
\log_{10}\gamma_i &= -\frac{Z_i^2\,A(T)\sqrt{I_o}}{1+\mathring{a}_i B(T)\sqrt{I_o}}
\end{aligned}
\tag{3}
$$

where m_i is the molar concentration of ion i, γ_i is the corresponding activity coefficient, $\mathring{a}_i$ is the effective diameter of the species in angstrom, Z_i is the valence of the i'th species or complex. $A(T)$ and $B(T)$ are functions that depend on the temperature (Helgeson, Kirkham et al. 1981):

$$\begin{aligned} A(T) = {} & 0.51 \cdot 10^{-3} - 1.154 \cdot 10^{-3}\,t + 35.697 \cdot 10^{-3}\,t^2 \\ & - 182.023 \cdot 10^{-3}\,t^3 + 346.528 \cdot 10^{-3}\,t^4\,, \\ B(T) = {} & 0.325 \cdot 10^{-3} + 0.08 \cdot 10^{-3}\,t + 1.441 \cdot 10^{-3}\,t^2 \\ & - 6.541 \cdot 10^{-3}\,t^3 + 11.655 \cdot 10^{-3}\,t^4\,, \end{aligned} \tag{4}$$

t is the temperature in °C, and I_o is the ionic strength given by:

$$I_o = \frac{1}{2}\sum_{i}^{N_b} Z_i^2\, m_i + \frac{1}{2}\sum_{i}^{N_c} Z_i^2\, n_i\,. \tag{5}$$

N_b and N_c is the number of basis species and complexes respectively. Usually the total concentrations of the basis species are known prior to any dissolution or precipitation, except for the H^+ concentration. The H^+ concentration can be found by requiring that the solution has no net charge i.e:

$$\sum_{i=1}^{N_b} Z_i\, m_i + \sum_{i=1}^{N_c} Z_i^c\, n_i = 0\,. \tag{6}$$

Conservation of the mass of a species is given by the following equation:

$$c_j = m_j + \sum_{i=1}^{N_c} \mu_{ij} n_i. \tag{7}$$

Mineral dissolution and precipitation is usually described by the following rate law:

$$\frac{\partial c_i(t)}{\partial t} = \sum_j \xi_{ij}\, I_j, \tag{8}$$

where A is the total surface area of the rock, and V is the bulk volume of the rock. We have used the following form of the flux I_j:

$$\begin{aligned} I_j &= \frac{A}{V} sgn(1-\Omega_j)\left(k_1' e^{-\frac{E_1}{R}\left(\frac{1}{T}-\frac{1}{T_{ref}}\right)} + k_2' e^{-\frac{E_2}{R}\left(\frac{1}{T}-\frac{1}{T_{ref}}\right)} a_H\right)\left|1-\Omega_j^m\right|^n, \\ &= sgn(1-\Omega_j)\left(k_1 e^{-\frac{E_1}{R}\left(\frac{1}{T}-\frac{1}{T_{ref}}\right)} + k_1 e^{-\frac{E_2}{R}\left(\frac{1}{T}-\frac{1}{T_{ref}}\right)} a_H\right)\left|1-\Omega_j^m\right|^n, \end{aligned} \tag{9}$$

Where the rate constants, k_1, and k_2 are redefined to include the factor A/V, to have the unit of mol/liter/second, $\xi_{ij}$is the stoichiometric matrix which relates the number of moles of a mineral to a basis species (e.g. when one mol of dolomite dissolves, one mol of calcium, one mol of magnesium and two moles of carbonate is released into the bulk solution) . Ω_j is the saturation index of mineral j, and is a function of the activity of the basis species. By discretizing equation (8) we can determine the change in the aqueous concentration of the basis species during one time step[1]:

$$c_i(t+\Delta t) - c_i(t) = \Delta t \sum_j \xi_{ij} I_j(t+\Delta t), \tag{10}$$

Combining this equation with equation (7) we find:

$$c_i(t) = m_i(t+\Delta t) + \sum_{i=1}^{N_c} \mu_{ij} n_i(t+\Delta t) - \Delta t \sum_j \xi_{ij} I_j(t+\Delta t), \tag{11}$$

The natural variables in this equation are the activities of the basis species, however to avoid negative activities in the Newton iterations, we change variables from $\boldsymbol{a}_i \rightarrow \log_{10} \boldsymbol{a}_i$.

Ion exchange and surface complexes

The next step is to add ion exchange reactions and surface complexes to equation (11), this is done quite simply by extending the set of complexes defined in (2) to also include complexation with a surface and introduce a new basis specie X^-, e.g.

$$\begin{gathered} Na - X \rightleftharpoons Na^+ + X^- \;, \\ Ca - X_2 \rightleftharpoons Ca^{2+} + 2X^- \,, \\ Mg - X_2 \rightleftharpoons Mg^{2+} + 2X^- \,. \end{gathered} \tag{12}$$

Surface complexes are also introduced in a similar way but then one must also introduce an equation that describes the conservation of charge, the Grahame equation (Israelachivili 1985)

$$\begin{aligned} &2 \cdot 10^3 \varepsilon \varepsilon_0 RT \sum_{i=1}^{N_b} m_i \left(E_0^{-Z_i} - 1\right) \\ &+ 2 \cdot 10^3 \varepsilon \varepsilon_0 RT \sum_{i=1}^{N_c} n_i \left(E_0^{-Z_i} - 1\right) - \sigma^2 = 0 \,, \end{aligned} \tag{13}$$

Where ε is the permittivity constant for a brine, calculated using the same approach as (Johnson, Oelkers et al. 1992), and ε_0 is the permittivity of vacuum. E_0 is a new basis specie representing the surface potential, $E_0 = e^{-F\psi/RT}$, where F is Faradays constant, R is the ideal gas constant and $T$is the absolute temperature, and σ is the surface potential defined as the sum of all the surface complexes

$$\sigma = \frac{F}{S} \left(\sum_{i=1}^{N_b} Z_{sci} m_i + \sum_{i=1}^{N_c} Z_{sci} n_i \right) . \tag{14}$$

For more details see (Hiorth, Cathles et al. 2010) and (Revil, Pezard et al. 1999).

Explicit Diffusive Layer Calculation

1 If we want to find the equilibrium solution (i.e. when$\Delta t \rightarrow \infty$ we do this by choosing dolomite and calcite as basis species instead of Mg, and Ca. This makes it possible to do equilibrium calculations in one step.

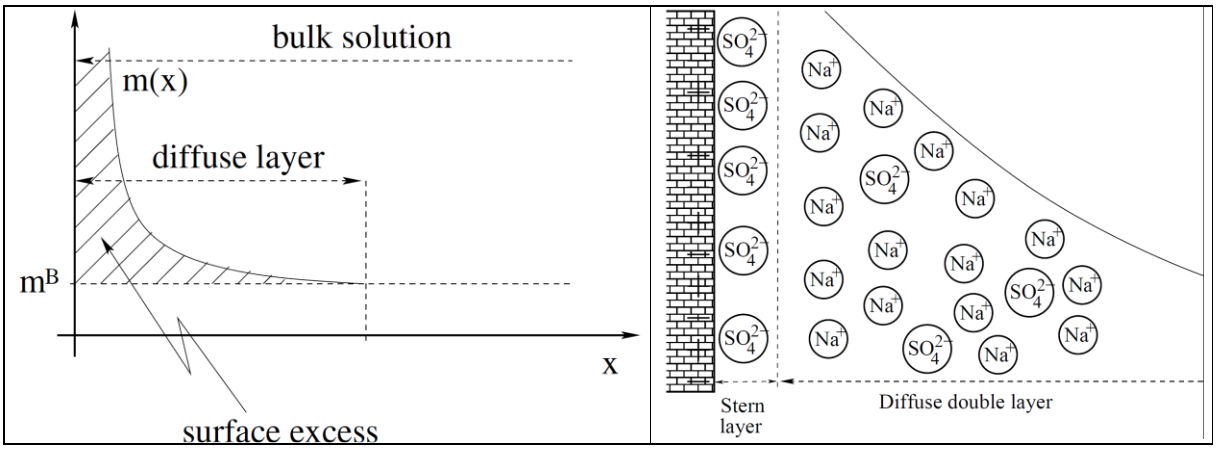


Figure 2: An illustration of the distribution of ions close to a charged surface, x is the distance from the surface and m^B is the bulk concentration (far from the charged surface) of an ion. If the surface is negatively charged a positive ion will have a higher concentration close top the surface. The right figure illustrates how a negative ion (sulphate) can adsorb onto the surface and change surface charge.

IORSim also has the ability to calculate the composition in the diffusive layer. We have implemented the model presented in (Borkovec and Westall 1983). In this model one calculates the composition of the diffusive layer close to a charged surface. The calculation is done by introducing the surface excess

$$\Gamma_i = \int_{X_d}^{\infty} m_i^B (e^{-Z_i F\psi(x)/RT} - 1)\mathrm{dx}. \tag{15}$$

Here m_i^B means the concentration of a basis species or a complex. X_d is the location of the outer Helmholtz plane (Borkovec and Westall 1983). The concentrations of ions in the diffusive layer can be calculated by multiplying the equation above with the specific surface area, S, of the rock (with the unit of m^2/liter)

$$m_i^\Gamma \equiv S\Gamma_i \equiv g_i m_i^B, \text{ where}$$
$$g_i \equiv -\frac{\sqrt{\kappa_s}}{4F} sgn(E_0 - 1) f_i, \tag{16}$$
$$f_i \equiv \int_1^{\frac{1}{E_0}} \frac{E^{-Z_i} - 1}{\left[E^2 \sum_i m_i^B (E^{-Z_i} - 1)\right]^{\frac{1}{2}}} dE\,.$$

Although the mathematics is slightly complicated, and one must evaluate the integral in equation(16) , the resulting equations fits nicely into the formulation we have already introduced. It turns out that the only modification we must do is to multiply all concentrations in equation (11) with $1 + g_i$. The Grahame equation defined in equation (13) is now replaced by

$$\sum_i Z_i m_i^\Gamma = \sigma, \tag{17}$$

Where σ is the surface charge defined in equation (14).

Algorithm for solving the geochemical system

In order to determine how the solution chemistry is changed during one time step, equation (2),(3),(5),(6), and (11) has to be satisfied[2]. In principle, all equations could be solved simultaneously using Newtons method, however by trial and error we have found that the most stable method is to solve these equations in two separate steps:

1. Equation (11) is solved by using Newtons method, but the ionic strength and activity coefficients are kept constant
2. When equation (11) has converged, the ionic strength and activities are updated and equation (11) is solved again until the ionic strength and activity coefficient do not change. If explicit diffusive layers are used, we update the integrals defined in equation (16).
3. If charge balance is a constraint e.g. when the H^+ concentration is unknown, the H^+ concentration is also updated by a secant step in the same loop where the ionic strength and activity coefficients are updated. This is usually done in the initialization step where H^+ is unknown, after the initialization step the total H^+ concentration is calculated and after that, we constrain the H^+ concentration by requiring mass balance (equation (11)).

One might think that an optimization to the code would be to include the ionic strength and activity coefficient dependency on the basis species in equation (11). This is rarely done, because the associated overhead when calculating the Jacobian makes this actually slower (Bethke 1996).

Geochemical Model in IORSim

In IORSim it is possible to define new interactions that are not initially present, this is done by including a file 'ions.txt' in the run catalogue. If the keyword HKF is given, the geochemical solver uses the HKF EOS to calculate the log K values. The units and the symbols are the same as those given in the appendix of (Johnson, Oelkers et al. 1992). If the keyword ANA is given, the geochemical solver calculates the log K directly based on the formula

$$\log K = A_1 + A_2 T + \frac{A_2}{T} + A_3 \log T + \frac{A_4}{T^2}. \tag{18}$$

$A_{1\ldots4}$ is given as input, if to few is given the rest is assumed to be zero. The input is quite flexible as illustrated below. The species are read in before the simulation is started and added to the data base, ions that are not used in the simulation are simply removed from the database.

```
#BASIS_SPECIES
#Name a0 Mw      DeltaG     DeltaH     S     a1    a2    a3    a4    c1
      c2    omega
#name a0 mol_weight (if ANA is used)
#ACETATE
ACE-  5 59.04 /HKF -88270   -116160    20.6 7.7525     8.6996
      7.5825     -3.1385    26.3  -3.86 1.3182     /
/end
SECONDARY_SPECIES
# DeltaG   DeltaH     S     a1    a2    a3    a4    c1    c2    omega
# ANA = a0 + a1 * T + a2/T + a3 * logT + a4/T^2
NaX = X- + Na+ / ANA 0. /
```

2 We have not included equations that describes surface complexation, this will basically yield one additional equation (the Grahamme equation), which relates surface charge to surface potential.

```
CaX2 = 2X- + Ca+2 / ANA -0.8 /
MgX2 = 2X- + Mg+2 / ANA -0.6 /
KX = X- + K+ / ANA -0.7 /
BaX2 = 2X- + Ba+2 / ANA -0.91 /
SrX2 = 2X- + Sr+2 / ANA -0.91 /
BgCl- = Bg+2 + 2.34Cl- / HKF -17450   -18270     34.1  5.2088
      4.9399     3.8015     -2.9832    9.8168     -1.6698    -0.03 /
GCO3Bg+ = GCO3- + Bg+2 / ANA .2 /
/end
EXCHANGE_SPECIES
#ion exchange species
#name
Z+
/end
SECONDARY_SPECIES
#dissociation of acetic acid
ACEH = ACE- + H+ /HKF -94760     -116100    42.7  11.6198    5.218
      2.5088     -2.9946    42.076     -1.5417    -0.15 /
#ion exchange with sulphate and chloride - not standard
SO4Z2 = 2Z+ + SO4-2 / ANA  -0.8 /
ZCl    = Z+ + Cl- /ANA -0.7/
/end
MINERAL_PHASES
# HKF = mol_volume    DeltaG     DeltaH     S     a1    a2    a3 a7
# ANA = mol_volume a0 + a1 * T + a2/T + a3 * logT + a4/T^2
WITHERITE = Ba+2 + HCO3- - H+ /HKF 45.81 -278400 -297500   26.8  21.5
      11.06 -3.91       /
#K-FELDPAR = Al+3 + 2H2O - 4H+ + K+ + 3SiO2 /HKF 108.87 -895374  -
949188     51.13 76.617    4.311 -29.945
/end
```

The '.geocheminp' file contains information about the rock, which minerals are present, amount of exchange sites and amount of surface complexes. Currently the specific surface area for the minerals (Sg) is not used, it enters in the parameter k_1 and k_2 (see equation (9))[3].

Table 2: Parameters in the rate keyword, the specific surface area is not used

Rate Parameter number	Parameter description	Symbol	Unit	Example value
1	Name of mineral	Text	No unit	Calcite
2	Amount present in rock		mol/kg rock	38
3	Specific surface area	SA	m^2/liter rock	1
4	Log activity of mineral	$\log a$	Dimensionless	0
5	Activation energy	E_1	J/mol	37.8e+3
6	Neutral rate constant	k_1	mol/s	3.43e-2

[3] The original idea was to include a specific surface area for each mineral and use this to calculate changes in permeability, in the same way as for the silicate model.

7	Activation energy	E_2	J/mol	8.4e+3
8	Acidic rate constant	k_2	mol/s	1.11E+03
9	Potens in rate constant	m	Dimensionless	1
10	Potens in rate constant	n	Dimensionless	1

Table 3: Parameters in the iexchange keyword, note that other exchange sites can be defined in the 'ions.txt' file

iexchange Parameter number	Parameter description	Symbol	Unit	Example value
1	Name of specie	Text	No unit	X
2	Amount present in rock		mol/kg rock	0.01

Table 4: Parameters in the complex keyword. If the size of diffusive layer is entered, extended diffusive layer calculations are used. These calculations are more time consuming and does not always converge

Line	complex Parameter number	Parameter description	Symbol	Unit	Example value
1	1	Numerical method	Text	No unit	method
1	**2**	Which method used	integer	No unit	1 or 2
2	**1**	Specific surface area	Text	No unit	**s_area**
2	**2**	Value of surface area	**SA**	mol/kg rock	1000
2	**3 (optional)**	Size of diffusive layer	**D**	nm	10
3	**1**	Name of surface complex	**Text**	No unit	GCa
3	**2**	amount of complex	**C**	Sites/nm^2	2

Below is an example of the keywords in the rate file.

```
rate
#    rate    =    (k_1*exp(-Ea/Rg)(1/T-1/298.15)+k_2*exp(-Ea/Rg)(1/T-
1/298.15)*(1-SI^n)^m
#mineral   mol/kg     Sg    log_af    logEa_1   k_1          logEa_2 k_2
             n  m
calcite    38.454     1     0     37.8e+3    3.43E-02    8.4e+3
      1.11E+03   1     1
magnesite  0.0    1     0     60e+3 .7E-08          0.0    0.0          1
      1
anhydrite  0.0    1     0     60e+3 7E-08      0.0    0.0          1     1
barite          0.0    2     0     30.8  1.11E-03    0.0    1.11E-03    1
      1
/end
```

```
iexchange
# mol/liter
X 0.01
/ end
complex
# method = 0 surface potential is set to zero
# method = 1 (default) solves Grahame equation
method 1
#specific surface area m^2/L pore volume
#if explicit diffusive layer calculation the size of diffusive layer
#in nm must be given
# s_area size of diffusion layer
# 1000 10
s_area 1000
#Name sites/nm^2
GCa 2
GCO3 2
/ end
```

The solution data are defined in input file < CASE>.trcinp. In the example below, two water solutions are defined by using the *SOLUTION keyword.

```
*solution solution0
   pH  9
   Ca  0.15
   Mg  0.021885
   Cl  2.5
   HCO3 1e-8
   Na  1.1428
   SO4  1e-8
   K   0.0073
   Ba  0.00184
   Sr  0.0085

*solution solution1
  pH      1.
  Ca      0.013
  Mg      0.0445
  Cl      0.525
  HCO3    0.002
  Na      0.45
  SO4     0.024
  K       0.01
  Ba      0
  Sr      0
```

After this, the keyword

*MODELTEMPLATE Comp1 (say)

defines a named area in the reservoir which is chemically initialized as follows:

The keyword

*COMP solution0

defines that IORSim will use solution0 as initial water composition for this region.

The keyword

*CHEMFILE ratefile

defines the name of a rate-file for this region (format for ratefile described above)

Each solution defined with keyword *SOLUTION, can also be defined as injected water composition for different wells in the reservoir.

The silicate model in IORSim

Theory

Sodium silicate polymerization is a complex process that is affected by several factors such as pH value temperature, silica concentration, and salt impurities. Stavland et al. (2011) conducted an experimental study to describe the gelation of sodium silicate by an empirical relation

$$t_{gel} = \xi e^{\alpha C_{Si}} e^{\beta C_{HCl}} e^{\gamma \sqrt{C_{Ca}}} e^{\frac{E_\alpha}{RT}} \qquad (19)$$

where C_{Si} is the concentration of Krystazil 40 (sodium silicate) in wt%, C_{HCl} is the hydrochloric acid concentration of 2M HCL solution in wt%, C_{Ca} is the solutions' calcium concentration in ppm and T is the temperature in Kelvin. The empirical parameters ξ = 8.75 10^{-10}, α = -0.6 1/wt %, β -0.7 1/wt %, and γ = -0.1 1/$\sqrt{ppm}$ were estimated during a series of sodium silicate polymerization experiments.

The numerical implementation (mass conversation) requires a reformulation of Equation (19) in which the concentration of sodium silicate is in balance with the sodium silicate gelation rate (Hiorth, Sagen et al. 2016). The sodium silicate polymerization/generation of the immobile gel species C_{gel} is proportional to the rate of sodium silica C_{Si} consumption

$$\frac{dC_{gel}}{dt} = -\frac{dC_{Si}}{dt} = kC_{Si}^n \qquad (20)$$

where a simple power-law describes the sodium silicate consumption using the variables k and n. Since the sum of $C_{Si}$and C_{gel} is constant, Equation (20) can be rewritten as

$$-\frac{1}{kC_{Si}^n} \cdot \frac{dC_{Si}}{dt} = 1 \qquad (21)$$

Integrating over time and replacing the integral variables by concentration leads to Equation (24)

$$\int_0^{t_{gel}} dt = \int_0^{t_{gel}} -\frac{1}{kC_{Si}^n} \cdot \frac{dC_{Si}}{dt} dt \qquad (22)$$

$$\int_{C_o}^{C} -\frac{1}{kC_{Si}^n} dC_{Si} = \left[\frac{C^{1-n}}{k(n-1)}\right]_{C_o}^{C} \qquad (23)$$

$$t_{gel} = \frac{C_0^{1-n}}{k(1-n)}\left[\left(\frac{C}{Co}\right)^{1-n} - 1\right] \qquad (24)$$

As the sum of the sodium silicate and immobile gel remains constant, C= $C_o - C_{gel}$ is assumed. Moreover, sodium silicate requires a minimum threshold concentration of X wt% to polymerize. Using 1/k = $\xi exp(\beta C_{HCl}) exp\left(\gamma\sqrt{C_{Ca}}\right) exp\left(\frac{E_\alpha}{RT}\right)$ as the kinetic reaction constant, a threshold sodium silicate concentration of X = 0.3 wt%, and a kinetic law exponent n of 4, Equation (19) and Equation (24) are combined to obtain a modified description of sodium silicate gelation.

$$t_{gel} = \xi e^{\beta C_{HCl}} e^{\gamma\sqrt{C_{Si}}} e^{\frac{E_\alpha}{RT}} \left[\frac{\alpha C_o^{1-n}}{1-n}\left[\left(1-\frac{X}{C_o}\right)^{1-n} - 1\right]\right] \tag{25}$$

Where α =100. Equation (25) ensures mass conservation by coupling the sodium silicate consumption to the sodium silicate gelation rate.

After defining the sodium silicate gelation, the impact of sodium silicate precipitation on permeability reduction is derived. Stavland et al. (2011) conducted coreflooding experiments to demonstrate that the injection of sodium silicate decreases the permeability in a range of 10^4 to 10^5. As sodium silicate is injected into porous media and polymerizes, the precipitated immobile silica reduces the available pore space occupied by the oil and water phase. A widely accepted mathematical description that relates porosity and permeability is the Kozeny-Carman model

$$k = \frac{\phi}{\alpha v \tau S_p^2} \tag{26}$$

where k is the permeability in Darcy, ϕ is porosity, α= 9.869 $^{-13}$ is a conversion factor to convert between Darcy and SI units, v is a shape factor, τ is tortuosity and S_p is the specific surface area per pore volume ($S_p = A_p/V_p$) in 1/m. Assuming that the sodium silicate polymerization occurs inside the pore throats, the tortuosity and porosity remain constant throughout the permeability reduction process. Consequently, the relation of initial permeability k_i and the modified permeability k_m becomes proportional to the relation of the initial specific surface area per pore volume $S_{p,i}$ and the modified specific surface area per pore volume $S_{p,m}$

$$\frac{k_m}{k_i} = \left(\frac{S_{p,i}}{S_{p,m}}\right)^2 = \left(\frac{1}{1 + Y S_w \frac{\rho_w}{\rho_{Si}} \frac{S_{Si}}{S_{p,i}}}\right)^2 \tag{27}$$

where Y is the concentration of the polymerized sodium silicate in fraction, ρ_w is the water density in kg/m^3 and ρ_{Si} is the sodium silicate density in kg/m^3. When assuming a spherical sodium silicate particle shape, the specific sodium silicate surface area is calculated by dividing the sodium silicate surface area ($4\pi r^2$) by the sodium silicate particle volume ($4/3\pi r^3$). Using Equation (26) to express $S_{p,i}$ and inserting into Equation leads to

$$\frac{k_m}{k_i} = \left(1 + \frac{3 S_w \rho_w}{r_{Si}\rho_{Si}} \sqrt{\frac{\alpha v \tau k_i}{\phi_i}}\right)^{-2} \tag{28}$$

Assuming τ =3, a geometrical shape factors v = 2, ρ_w as 1000 kg/m^3 and ρ_{Si} as 2650 kg/m^3, and sodium silicate particle radius of 10 nm, the final permeability expression yields into

$$k_m = \left(1 + 275.5\, Y\, S_w \sqrt{\frac{k_i}{\Phi_i}}\right)^{-2} k_i \qquad (29)$$

Sodium silicate model in IORSim

Sodium silicate model implementation in IORSim

IORSim uses five species to ensure the implementation of numerical sodium silicate simulations: Sodium silicate, hydrochloric acid, calcium, immobile gel, and mobile gel.

- Sodium silicate, hydrochloric acid and calcium are dissolved species inside the water phase
- The mobile gel species reflects the immediate generation of sodium silicate aggregates. The nanosize aggregates can be transported a significant distance within the reservoir.
- The immobile gel species represents the sodium silicate that precipitated inside the porous network and reduces the permeability.

```
#-- Selection of IORSim model -> here Silicate
*MODELTYPE SILICATE
#-- Enummertatiom of species
*SPECIES Silica
*SPECIES MobGel
*SPECIES HCl
*SPECIES Ca
*SPECIES ImMobGel
```

While IORSim uses the host simulator to specify and calculate the well injection rates, IORSim defines the species concentrations. The below listed example initially assumes the injection of fresh water in which only calcium is present (200ppm). After 31 days, 4 wt% sodium silicate, 3 wt% HCL, and 20 PPM calcium are injected. In both cases, an injection temperature of 25°C is assumed.

```
#-- Wellname
Inj
#-- NtimE
```

```
2
#--Time C_Si C_MobGel C_HCL C_Ca C_ImMobGel Injection_temperature
 0 0 0 0 200 0 25 # Injection of water (0 days, 200 ppm Ca at 25C)
31 4 0 3 20 0 25 #Injection of Si and HCl (31 days,4 wt% Si, 3wt% HCl,
20 ppm Ca at 25C)
```

Sodium silicate model implementation in Eclipse

IORSim calculates the impact of sodium silicate precipitation on fluid flow by interpolating between different SATNUM sets. By default, IORSim uses 12 SATNUM input sets. Consequently, the keywords SWOF (water/oil saturation functions versus water saturation, SGOF (gas/oil saturation functions versus gas saturation), and SPECROCK (Rock specific heat data) need to be defined by an equal amount (12) of input sets. To allow the processing of 12 relative permeability/capillary pressure curves, the TABDIMS (table dimensions) specifications need to be adapted.

Sodium silicate illustration example

Figure to Figure illustrate the injection of sodium silicate into an idealized reservoir. While the injection of water leads to an efficient oil recovery from the high permeable zones, the impermeable layers remain unwept (Figure). Once sodium silicate is injected into the reservoir, the sodium silicate accumulates inside the high permeable layers (Figure). The injected sodium silicate precipitates around the temperature front, as displayed in Figure . As a result of the sodium silicate precipitation, the permeability decreases (Figure). Consequently, the injected water imbibes into the low permeable zones, displaces the oil, and increases the oil production (Figure).

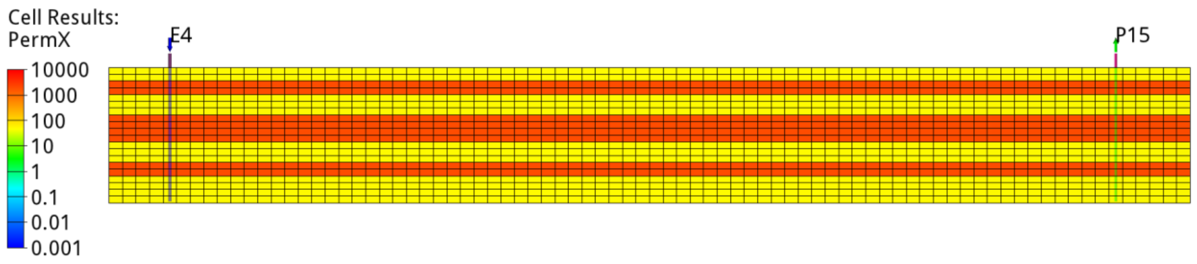


Figure 1: Initial permeability distribution. A layered reservoir model with high permeable zones (2000mD)

and low permeable zones (50 mD).

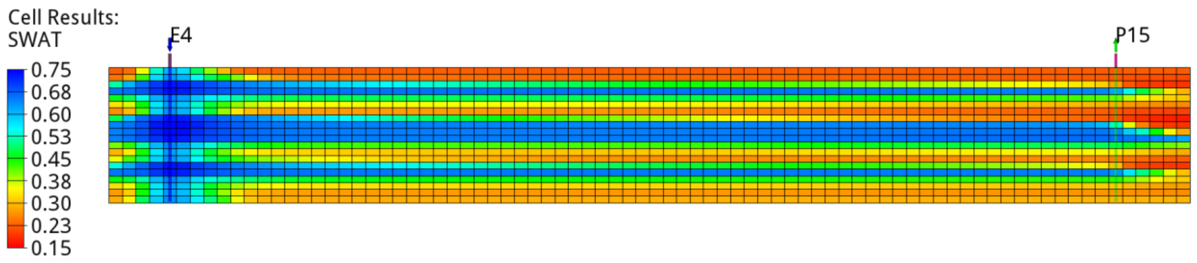


Figure 2: Water saturation distribution before silicate injection. The layered reservoir properties lead to an efficient oil recovery from the high permeable zones while the less permeable zones remain unwept.

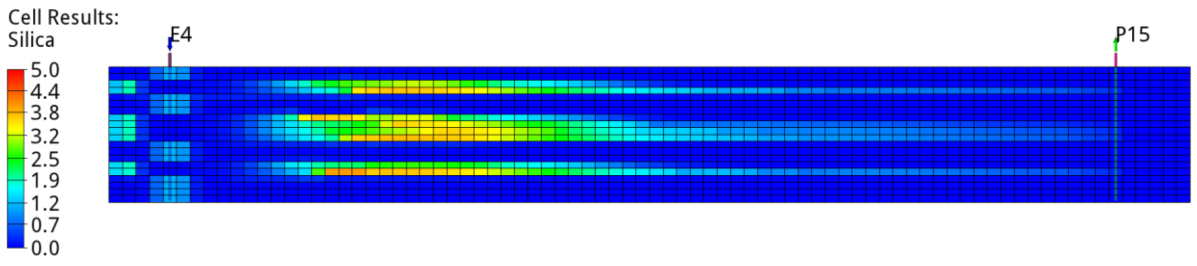


Figure 3: Silicate injection into the reservoir.
The injected silicate accumulates inside the high permeable layers

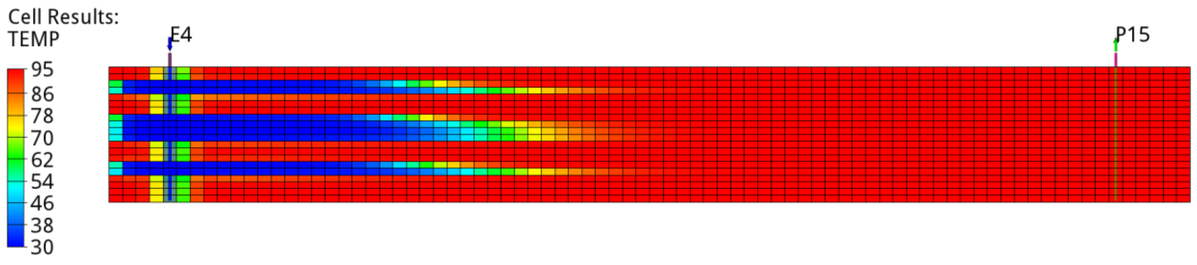


Figure 4: Temperature distribution. As a result of the water injection, the temperature decreases in the vicinity of the injection well.

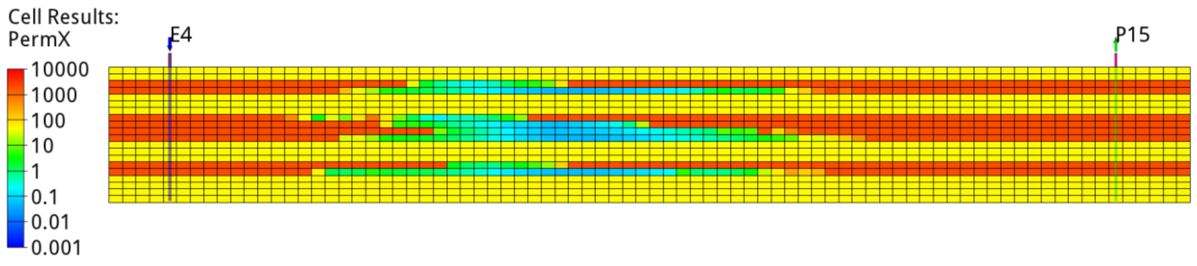


Figure 5: Sodium silicate precipitation. The injected sodium silicate precipitates around the temperature front.

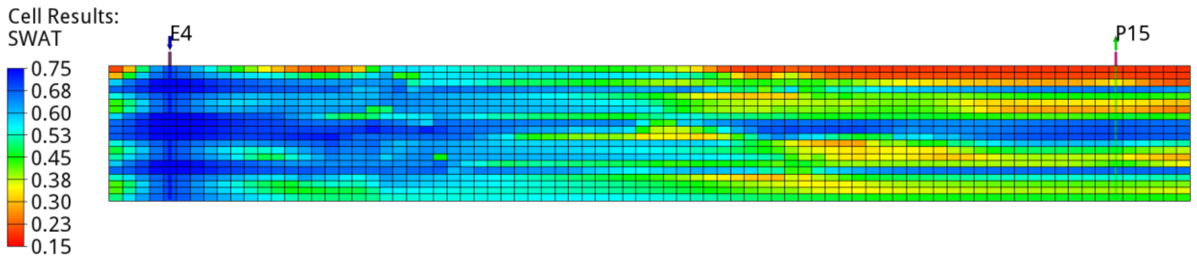
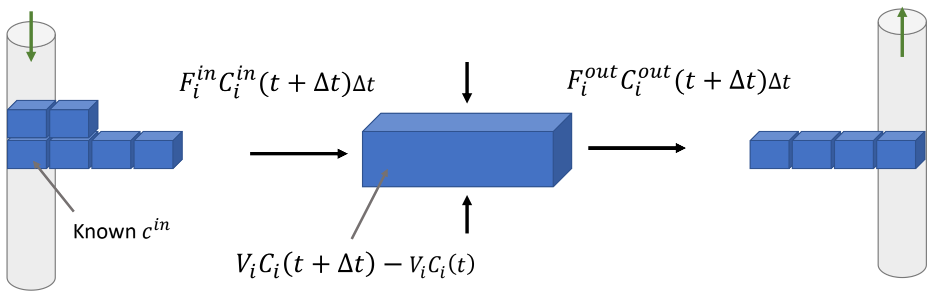


Figure 6: Water saturation distribution after silicate injection. The precipitated sodium silicate reduces the permeability and causes water diversion. The impermeable zone layers are swept.

Numerical methods

Since the geochemical calculations are separated from the host simulator, IORSim reads the Eclipse output files to construct an segregated cell structure. Static grid properties such as grid dimension, cell sizes, cell transmissibilities, permeability, and porosity are retrieved from root.egrid and root.init. Moreover, the global fluid flow (phase volume flow, formation factors, well flow rates, phase saturation, and phase volume) is read at each timestep from root.rft and root.unrst. While the files mentioned above are standard Eclipse output files, IORSim requires the creation of one additional input file. Depending on the selected geochemical module, root.trcinp defines the initial and injected species concentrations. Once IORSim has calculated the advective species transport and geochemical reactions (dissolution, precipitation, surface charge change, etc.), the impact of geochemical reactions on the fluid flow is communicated via updated relative permeability and capillary pressure curves (root.satnum). Depending on the magnitude of chemical reactions, IORSim assigns each Eclipse cell a Satnum value closest to predefined relative permeability/capillary pressure input data. The iterative process of starting and pausing the Eclipse and IORSim simulation is continued until the simulation is completed. After the completion of the simulation, the IORSim results are merged into root.egrid to allowing post processing programs such as Eclipse FloViz and ResInsight to be used.

Fully implicit reservoir simulators typically use a Newton-Raphson solver, in which the primary variables are stored inside a matrix and simultaneously solved over iterations. While the fully implicit numerical scheme is stable and tolerates large time steps, the handling of the global matrix requires high computational effort. In addition to the chemical species transport calculations, IORSim computes non-linear geochemical reactions over iterations at each time step. Compared to conventional flow simulation applications, reactive transport problems cause significantly larger computational times and convergence problems. To avoid handling an unstable global matrix, IORSim divides the reactive species balance calculations into a sequence of local systems. The principle of the approach is sketched in Figure 8.



$F_i^{in}C_i^{in}(t+\Delta t)\Delta t$
$F_i^{out}C_i^{out}(t+\Delta t)\Delta t$
Known c^{in}
$V_iC_i(t+\Delta t) - V_iC_i(t)$

Figure 8. Species balance equation in IORSim.

Before starting the species balance calculations, IORSim reads the host simulation results to detect the flow pattern. Once the flow paths are decrypted, the neighbouring cells are aligned into a sequence of upstream and downstream cells. The arising cell alignment is similar to a streamline simulation approach, in which the species balance solution is divided into an iterative string of 1D problems. As the flow pattern is detected, the sorting algorithm loops through the flow pattern to identify the injection cells. Starting from the injection cells, where the species concentrations are defined through boundary conditions, the species balance from a timestep *n* to a timestep *n+1* can then be expressed as following

$$V_i^n C_i^n + F_{in} C_{in}^{n+1} \Delta t = V_i^{n+1} C_i^{n+1} + F_{out} C_i^{n+1} \Delta t, \tag{30}$$

where V denotes the total fluid volume, $\mathrm{c_i}$ is the cell species concentration, F_{in} is the total volume influx from the neighboring (upstream) cells of cell i, $\mathrm{c_{in}}$ is the average species concentration of the upstream cells, and F_{out} is the outflux from grid cell i. The species balance notation neglects a source/sink term for simplicity.

As the cell influx F_{in}cell outflux F_{out}, and species influx concentration $\mathrm{c_{in}}$ are known, the only unknown C_i^{n+1} could be directly solved from Equation (30). However, as temporal and spatial discretizations tend to introduce mass balance errors, IORSim takes the volume balance of the host simulator into account

$$\mathrm{V_i^{n+1}} + \mathrm{F_{out}}\Delta t = \mathrm{V_i^n} + \mathrm{F_{in}}\Delta t \tag{31}$$

where the cell outflow term F_{out} is eliminated by inserting Equation (31) into Equation (30)

$$(\mathrm{V_i^n} + \mathrm{F_{in}\Delta t})\mathrm{C_1^{n+1}} = \mathrm{V_i^n C_i^n} + \mathrm{F_{in} C_{in}^{n+1}}\Delta t. \tag{32}$$

The purpose of IORSim is the calculation and visualization of geochemical reactions. Equation (32) is therefore extended by coupling a geochemical reaction term to the species balance

$$C_i^{n+1} = \frac{V_i^n C_i^n + F_{in} C_{in}^{n+1}\Delta t}{V_i^n + F_{in}\Delta t} \underbrace{-\left[\Delta t^{n+1}\,(1-\epsilon)V_i^n r(C_i^n) + \Delta t^{n+1}\,\epsilon V_i^{n+1} r\left(C_i^{n+1}\right)\right]}_{\text{geochemical reaction term}} \tag{33}$$

where ϵ is $\in$ [0,1] and $r_i(C_i)$ denotes non-linear geochemical reactions such as dissolution, precipitation and/or other chemical reactions.

Equation (33) is sequentially calculated for the upstream cells. The species transport is solved implicitly without calling a linear solver routine, whereas the geochemical reactions terms are stored inside a local cell matrix and solved over iterations. Besides a fully implicit numerical scheme, IORSim integrates an explicit numerical scheme and an adaptive numerical scheme. The numerical scheme is controlled by the Courant Friedrichs Lewy (CFL) condition, which is defined as

$$CFL = \frac{v\,\Delta t}{x}, \tag{34}$$

where v is the flow velocity, Δt is the time step size, and Δt is the cell length. The CFL condition states that the distance, which a fluid moves during one timestep, must be shorter than the length of the cells to obtain a stable explicit solution. Consequently, a CFL maximum value of 1 defines the threshold value until explicit schemes converge, while implicit schemes tolerate larger CFL values. Although IORSim utilizes a CLF maximum value of zero to achieve a fully implicit numerical scheme, the IORSim user can define an arbitrary CFL threshold value until an explicit scheme is applied. In the case of the explicit numerical scheme, the reactive species balance equation can be summarized as follows

$$C_i^{n+1} = \frac{\left(V_i^{n+1} - \Delta t F_{in}\right)C_i^n + \Delta t F_{in} C_{in}^n}{V_i^n + F_{in}\Delta t} - \left[\Delta t^{n+1}\,(1-\epsilon)V_i^n r(C_i^n) + \Delta t^n\,\epsilon V_i^n r\left(C_i^{n+1}\right)\right]. \tag{35}$$

The sequential solution method in IORSim

Background and need

The figure below illustrates a one-layer reservoir simulation case. The red border is the outer border of the reservoir. The black borders indicate the flow connections between each grid cell. The arrows indicate the direction and the size of the flow rates between two neighbour cells. The reservoir is seen from above, and the position of the wells is indicated by red dots.

IORSim solves the species concentrations in each cell at each time step of the simulation. Each cell has been initialized with concentration values, and at each time point in the simulation, a certain species concentration is injected into the well cells from the wells (red dot).

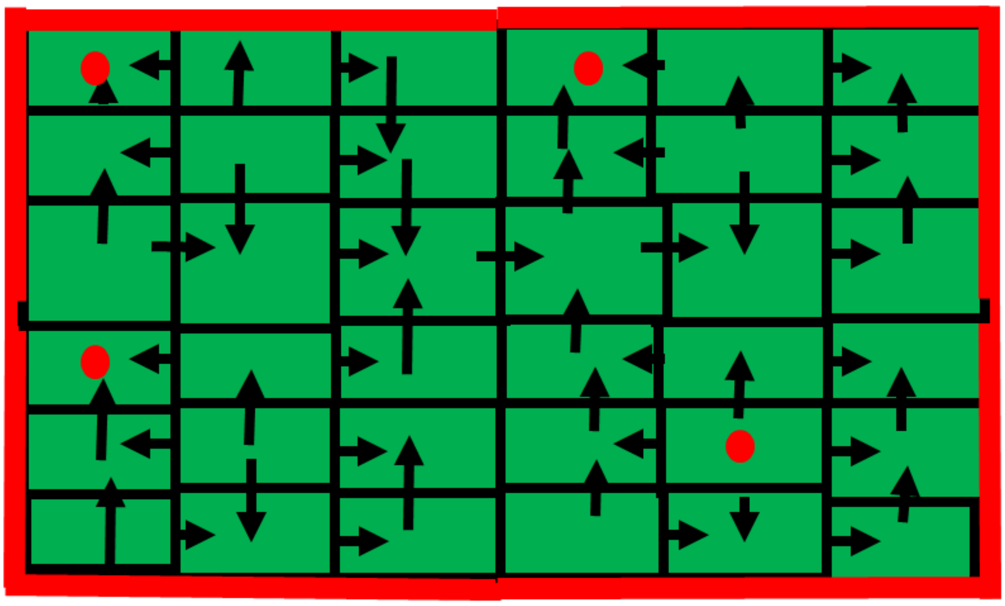


Figure 9. A schematic sketch related to solving the species balance equations in IORSim.

In addition to solving the transport equations of species across each connection of two and two grid blocks, the geochemical equations must also be included. The resulting system of equations is a coupled system of all grid blocks, where coupling to the non-linear geochemical equations is performed at each grid cell. The "normal" way of solving such a system, would be to put everything up in a huge matrix with lots of off-diagonal elements, and solve at each time step. Since the geochemical equations are non-linear, several iterations of this matrix solution would have to be performed. For such a large coupled non-linear system of equations, the CPU time would be huge, and in addition there is great risk of having convergence problems.

In the next section, we shall see that a much more practical and efficient method exists for solving this set of equations, introducing a so-called sorting algorithm for the grid cells.

Sorting algorithm for water species

The sequential method principle applied in IORSim is both simple and very computationally efficient. It is based on "localizing" the solution problem, instead of solving the total coupled non-linear equation system of cell concentrations simultaneously.

To understand the solution principle, it is useful to look at the "localized" problem, illustrated in the figure below. In the figure, F_{in} is the total water inflow for this grid cell, c_{in} is the average species concentration for all the inflow elements. The inflow elements consist of all well inflow completions connected to this grid cell, plus all the inflow connections from the neighbour grid cells. The outflows F_{out} are similarly defined. All the *Fin* s and F_{out} s are known from Eclipse. If we assume that c_{in} is also known, we observe that there is only one unknown quantity in this system, namely the unknown local concentration c_i. Hence this local system consists of only one unknown variable c_i, and the resulting non-linear equation is quite straightforward to solve efficiently by iteration (geochemical module). Now the question arises, can the c_{in} s be assumed to be known in advance? The answer is yes, if we have sorted the grid cells in in such a way that all the c_{in} s have been solved first.

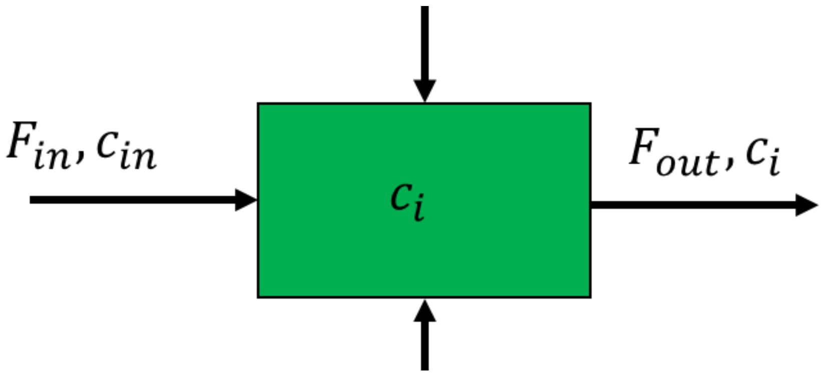


Figure 10. The mass balance of one grid cell in IORSim.

To achieve this at a certain time step, we loop through all the flows and the belonging pair of grid cells interconnected by this flow. Dependent on the flow direction for this connection, we interchange the sequence of the two blocks so that the upstream block relative to the flow, is listed before the downstream block. This procedure is repeated for all the flow connections in the reservoir. As a result, we obtain a grid cell sequence with the property that all inflow cells relative to a specific cell, are listed before the cell itself. See figure below.

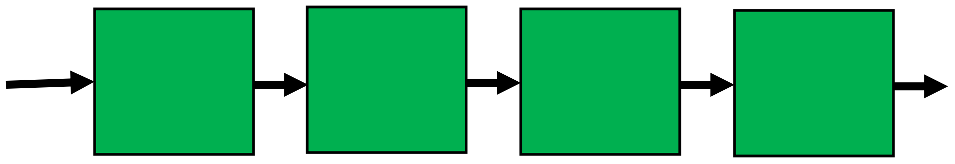


Figure 11. Solving a sequence of grid cell in IORSim.

As seen in the figure above, we obtain a one-dimensional sequence of cells, where all the flows are pointing in one direction, right, say. If we now start solving the localized problem at the outmost left side, we see that this cell can be easily solved, since the only inflow here must be coming from in injection well, and hence is a known quantity. Going to the next cell, we similarly see that it also can be solved, since the inflow from the left side was just calculated as outflow from the previous cell. In that way, we can continue trespassing from left to right, until all the grid cells are solved.

To illustrate that the method is valid for 2D and 3D cases, we choose a quarter of five-spot reservoir as an example. In the figure below, the blocks have been enumerated in a natural way, but the sorting algorithm works no matter which enumeration is chosen. We observe that the cell sequence
1, 2, 3, 4, 5, 6, 7, 8, 9 does the job of having inflow cells being calculated before the cell itself. We also observe that the sequence 1,4, 7, 2, 5, 8, 3, 6, 9 does the job.
The algorithm will pick one of them, dependent on how the cell-connection data structure has been set up

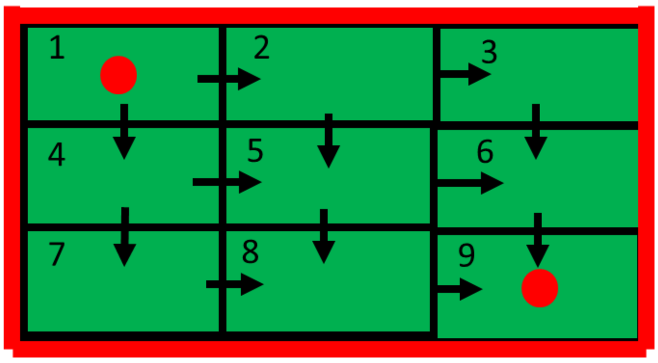


Figure 12. A quarter of fivespot example of a grid cell sequence in IORSim.

We now look at the case with reversed flow directions:

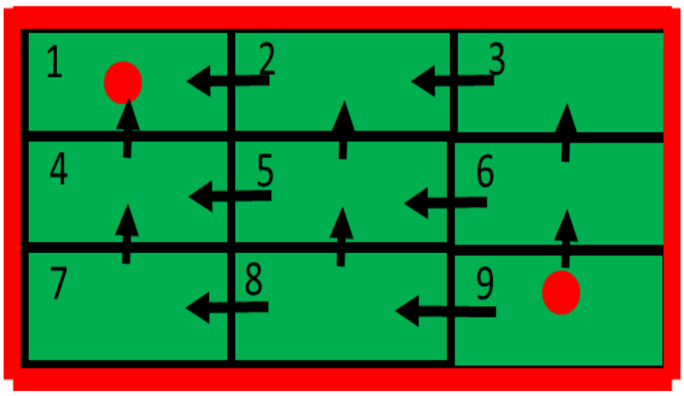


Figure 13. Reversing the flow

In this case the sequence 9, 8, 7, 6, 5, 4, 3, 2, 1 and the sequence 9, 6, 3, 8, 5, 2, 7, 4, 1 do the job.

Sorting algorithm for partitioning species

The sequential method for water species has been extended to handle species which exist in all three phases water, oil and gas. In the previous section, we described how the sorting algorithm tests on the direction of the water phase and sort the grid cells such that inflow cells are solved before the downstream cell. If the water and oil phases flow in different directions, we must choose which phase we take into account with respect to inflow. If we sort the cells according to the water phase, this means that the water flow terms are taken implicitly as before (inflow terms calculated first). Since water inflow cells are calculated first, and water and oil flow in opposite directions, the oil inflow cells will be calculated last. This again means that the inflow oil terms must be calculated using the old concentrations, i.e. integrated explicitly in numerical sense. The water flow terms are integrated implicitly (new concentration values) as before. Since we now have deduced that one of the phases must be handled explicitly in the case that water and oil are flowing in opposite directions, the question about numerical stability now arises. It is clear that such a solution can become numerically unstable, especially if the flow of the countercurrent phase is huge.

Actually, further stability analysis shows that the total solution is stable provided the countercurrent phase flow is not too big. The solution is numerically stable when:

$$\Delta t \frac{Fw-K*Fo}{(s+(1-s)*K)} \geq -V \qquad (36)$$

where Fw is water rate, Fo is oil rate, K is the K-value between the phases for this specie. This shows that when Fw > K*Fo, sorting should be done by the water inflow, else it should be done by oil inflow.

Separate grid refinement

In IORSim, the tracer calculation is performed on a separate grid which may be finer than the original Eclipse grid. The grid is shortly denoted a “tracer refined grid”. The purpose of the tracer grid is merely numerical. Since the tracer pulses in well-to-well tracer tests are quite narrow, it is very important to have a good numerical resolution of the pulses propagating through the reservoir. In fact, one application of the tracer grid refinement option is to obtain an exact solution of the convection-diffusion equation. By entering a specific value of the physical diffusion in the input file, the user may refine the tracer grid until the simulated tracer pulse no longer is changed when the grid is refined. If a zero value is entered for the physical dispersion, the solution will always become sharper when the grid is refined, meaning that it is not possible to strictly obtain a unique solution. This is actually the normal procedure in reservoir simulation, where usually no physical dispersion is specified, and a fixed number of grid blocks is applied. The shape of the tracer pulse is in this case determined only by the numerical dispersion corresponding to the chosen grid size. Below, the method of tracer grid refinement is described.

Description

```
N_lgr
I1 I2 J1 J2 K1 K2 NXref NYref NZref
. . . . . . . . .
. . . . . . . . .
I1 I2 J1 J2 K1 K2 NXref NYref NZref
```

This defines species grid refinement. The first argument is an integer that defines the number of areas that will be refined. If the first argument is 0, no grid refinement is performed.

If the first argument (N_lgr) is a positive integer, N_lgr x 9 integers must be given. In these groups of 9 integers, the six first (I1, I2, J1, J2, K1, K2) denote the boundary of the refined block, with lower and upper x-direction grid blocks given first, then lower and upper y-direction grid blocks and lower and upper z-direction grid blocks given at last. The last three numbers in the group of 9 integers denote the degree of refinement in x-direction, y-direction and z- direction, respectively. The last three integers must be odd numbers (NXref, NYref, NZref = 1,3,5,..). A number 1 gives the original grid block size, i.e. no refinement.

Adjacent grid refinement regions are allowed, e.g. cases as illustrated in Figure 1a and b. The adjacent regions can have equal degree of refinement or different degrees of refinement.

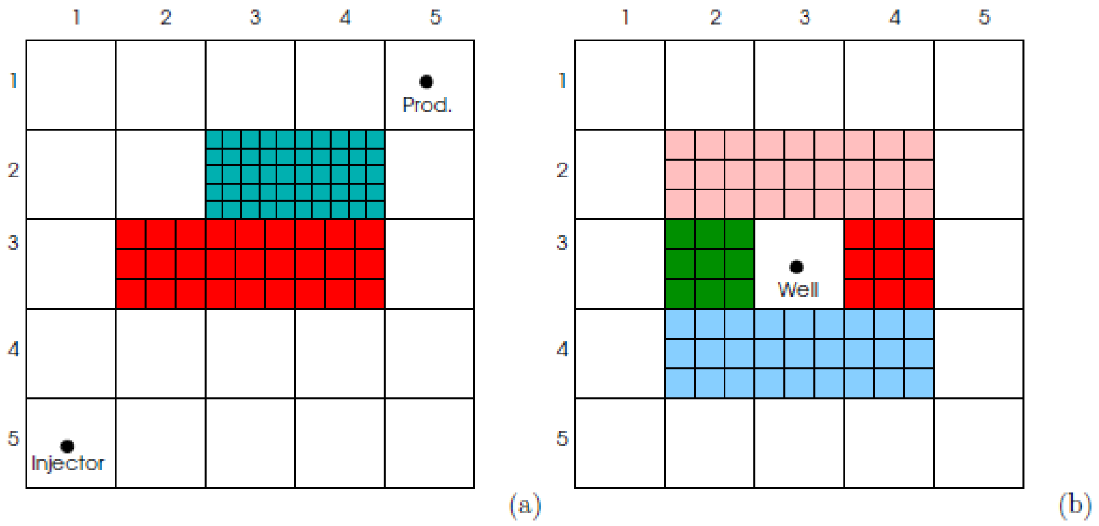


Figure 14 Grid refinement in IORSim. Adjacent independently refined regions are allowed

Velocity calculation between refined cells

Figure 1.1 illustrates an Eclipse well block. The red arrows represent the original flow rates read from Eclipse. The red filled circle in the middle of the block, represents the well completion flow rate, positive sign for a producer, negative sign for an injector.

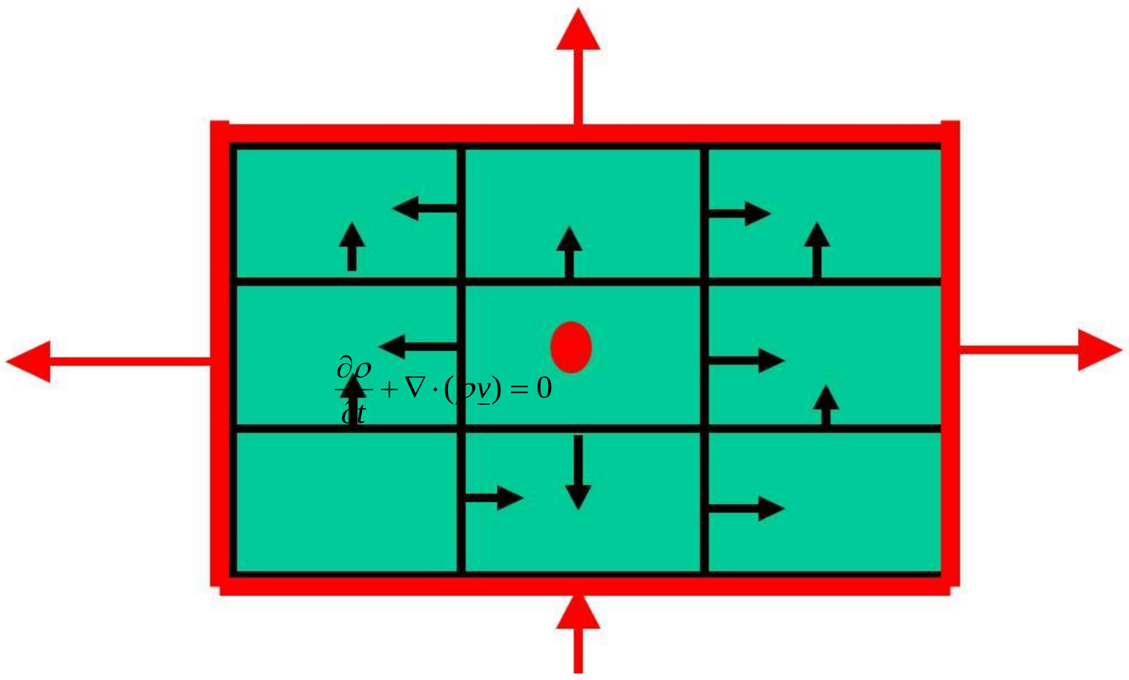


Figure 15. Black arrows: Refined velocities within the grid cell. Red arrows: Velocities in Eclipse

The figure above illustrates a well grid block in Eclipse. The red arrows indicate the flowrates of Eclipse, the black arrows indicate the tracer refined velocities (flowrates)

The black arrows are the calculated tracer refined velocities. We will show how the these refined velocities (flow rates) are calculated based on the original flow field read from Eclipse. The calculation is done separately for each of the three phases, water, oil and gas. Assume that we want to do the calculation at a specific time point in the Eclipse simulation. At this time point there is a specific build-up or build-down of net mass in the block, depending on the difference between the flow rates going into the block (red arrows), and the completion flow rate (red filled circle) going out (assumed positive flow). For each phase and at any point in space within the block, the following mass conservation equation applies:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \underline{v}) = 0$$

At this specific point in time, $\frac{\partial \rho}{\partial t}$ has a certain value dependent on the compressibility of the fluid at this time. Also, we assume that the compressibility build-up of mass is evenly distributed within the Eclipse block. Hence may be assumed constant. Likewise, we may assume that the mass is equally spaced within the block itself, so that ρ is only a function of time. Hence we may write:

$$\frac{1}{\rho}\frac{\partial \rho}{\partial t} + \nabla \cdot \underline{v} = 0$$

Now, we investigate the compressibility term further to quantify it based on the information Eclipse has supplied for this specific well block. On the one hand, the mass build-up in the block must be:

$$\frac{dM}{dt} = V\frac{\partial \rho}{\partial t}$$

On the other hand, by considering the total mass balance in the block, we must have:

$$\frac{dM}{dt} = \rho(Q - \sum_i Q_i)$$

The compressibility may be expressed:

$$\frac{1}{\rho}\frac{\partial \rho}{\partial t} = \frac{Q - \sum_i Q_i}{V}$$

The velocity equation to be solved becomes:

$$\nabla \cdot \underline{v} + \frac{Q - \sum_i Q_i}{V} = 0$$

By introducing the permeability *K* within the block and neglecting the viscosity which is assumed to be constant, we obtain:

$$\underline{v} = -\nabla \cdot (K\nabla P)$$

$$\nabla \cdot (K\nabla P) = \frac{Q - \sum_i Q_i}{V}$$

In IORSim, the pressure equation is first solved. Finally, the velocities are retrieved. Note that K is dependent on the local direction of integration. In IORSim Kx, Ky and Kz are indirectly calculated by applying the Tx, Ty and Tz transmissibilities, which are retrieved directly from Eclipse.

The vertical and horizontal well situations are illustrated in the figures below:

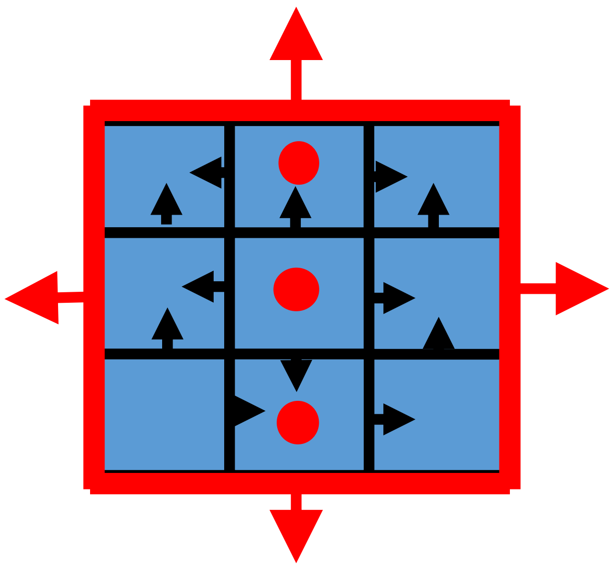


Figure 16. Refining a well block for a vertical well

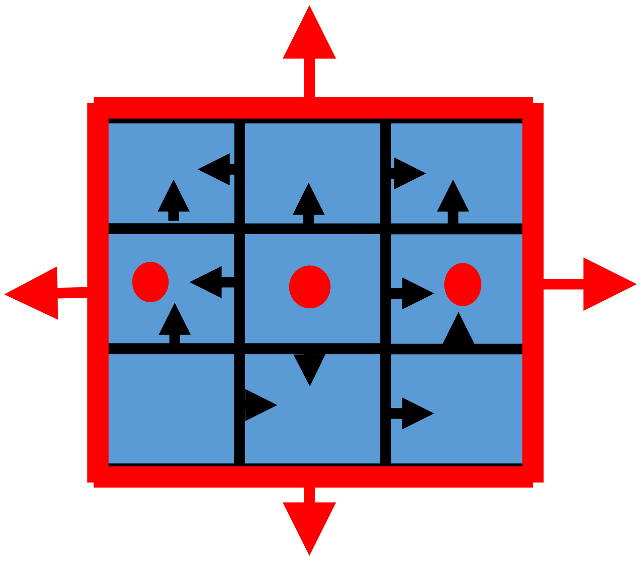


Figure 17. Refining a well block for a horizontal well

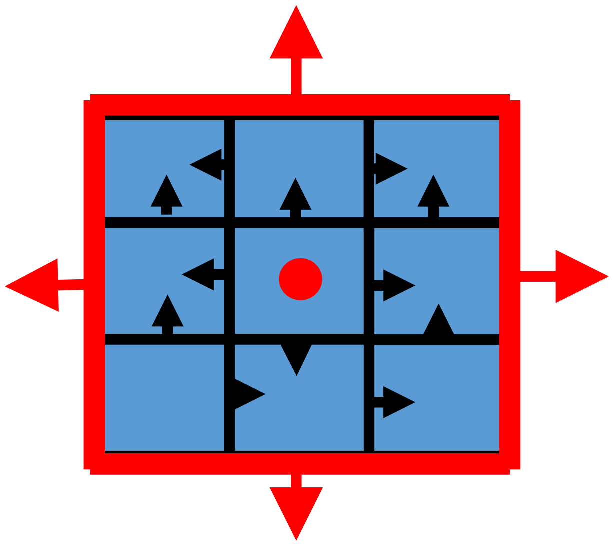


Figure 18. Refining a well block for a horizontal well

Below, a real well situation is iillustrated, together with a subsequent grid refinement realization.

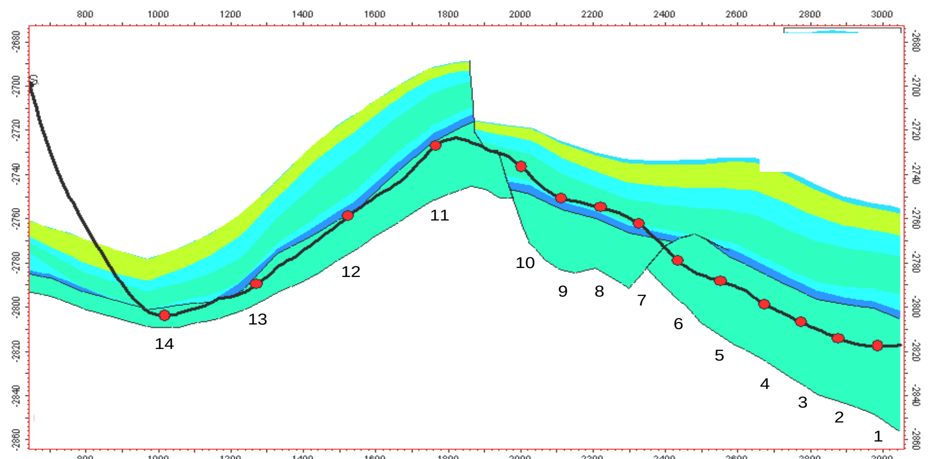


Figure 19. Sketch of a real horizontal well

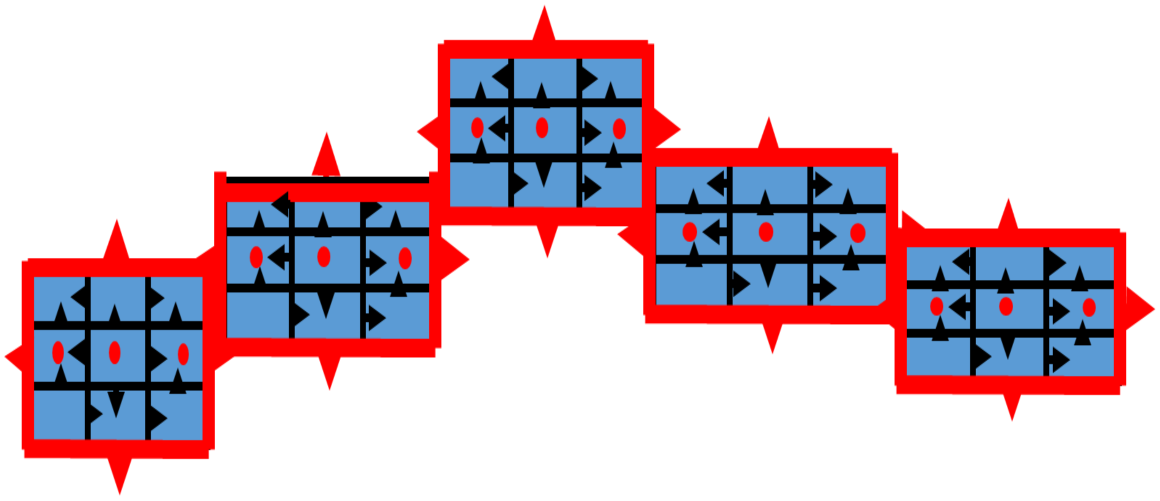


Figure 20. The grid representation of the horizontal well

Ideal tracers and partitioning tracers

By decoupling the tracer and fluid flow it is possible to state and solve the tracer problem based on previously solved and stored reservoir simulator runs. This approach allows for fast tracer simulations with execution times of 1-5% of the corresponding reservoir simulation, and the possibility to re-state and solve the tracer problem without re-solving the flow. Multiple tracer scenarios can therefore be simulated very fast. The sequential method for water species has been extended to handle species which exist in all three phases (water, oil, gas). For each specie, the distribution between the phases is governed by an equilibrium relation.

If we assume that partitioning among the phases is an instantaneous process a conservation equation for an arbitrary partitioning tracer component q may be written as

$$\frac{\partial}{\partial t}\left(\sum_{i=o,g,w} \varphi\, S_i\, K_i^q C^q\right) + \nabla\cdot\left(\sum_{i=o,g,w} \boldsymbol{v}_i K_i^q C^q\right) - \nabla\cdot\left(\sum_{i=o,g,w} \varphi S_i \boldsymbol{D}_i^* \cdot \nabla\left(K_i^q C^q\right)\right) = 0 \tag{37}$$

Here, partitioning is described by the coefficient $K_i^q = C_i^q / C^q$, where C_i^q is the concentration of q in phase i and C^q is the concentration in the primary phase. C^q is the primary variable solved for in the equations.

Neglecting the dispersion term in the above equation and simplifying the notation a little bit, we obtain the discretized tracer equation, actually solved in IORSim.

The accumulation term in the discretized equation becomes:

$$m_{tot} = V_{cell}(K_w\, \mathrm{C\ Sw}\, \varphi\ + K_o \mathrm{C\ So}\, \varphi + K_g \mathrm{C\ Sg}\, \varphi) \tag{38}$$

where K_w, K_o, K_g are the K-values for each phase for this specific specie. Note that the K-values in general can be functions of temperature, pressure and also composition. C is the primary concentration, which is solved for numerically in IORSim. The actual definition of K_w, K_o, K_g will depend on the definition of the primary concentration C (which phase etc.) C is calculated for all grid cells at each time step of the simulation.

Note that the equilibrium of tracers (and in general species) between the phases, is assumed to be instantaneous. The K-values Kw, Ko, Kg may be dependent on P, T and composition. Partitioning species in IORSim may be useful for instance when CO2 is present in water, oil and gas. When for instance Ko = Kg = 0, the tracer (specie) is ideal (also denoted passive) and sticks only to the water phase. Other ideal tracers may be pure oil tracers (Kw = Kg = 0) or gas tracers (Kw = Ko = 0).

The relations between C and each phase concentrations are

$$C_w = K_w\, C \qquad C_o = K_o\, C \qquad C_g = K_g\, C \tag{39}$$

For each grid cell, the total transport equation for the species then becomes:

$$\frac{\Delta m_{tot}}{\Delta t} = \frac{\Delta(V_{cell} K_w\, \mathrm{C\ Sw}\, \varphi\ + V_{cell} K_o \mathrm{C\ So}\, \varphi + V_{cell} K_g \mathrm{C\ Sg}\, \varphi)}{\Delta t} =$$

$$\sum_{\substack{flow \\ connections}} \left(K_{w,jup} C_{jup}\, Q_{w,jup} + K_{w,jup} C_{jup}\, Q_{w,jup} + K_{w,jup} C_{jup}\, Q_{w,jup}\right) \tag{40}$$

The last term is summed through all grid cell boundaries (flow connections). The grid cell jup is the upstream grid cell relative to the connection. This mass conservation equation is solved implicitly in IORSim.

Single well chemical tracers used for SWCTT tests

Single-well chemical tracer tests are based on injection of an ester into the reservoir. Some of the ester hydrolyses during a shut-in period, and subsequent production of the ester and the alcohol produced during shut-in yield tracer production curves that can be used to find residual oil saturation. Commonly utilized esters in SWCTT tests are propyl formate and ethyl acetate. Symbolically we can write the hydrolysis reaction as

$$\underset{\text{(ester)}}{R_1COOR_2} + \underset{\text{(water)}}{H_2O} \rightleftarrows \underset{\text{(alcohol)}}{R_2OH} + \underset{\text{(acid)}}{R_1COOH} \tag{41}$$

Ester is typically a partitioning tracer and alcohol is typically an ideal (passive) tracer. By just injecting and back producing ester, or even adding an ideal tracer, would only result in a back production of the ester and the ideal tracer at the same time, because of injection and back-production in the same production well. The clue of the SWCTT method is that alcohol (ideal tracer) is produced in the chemical reaction during the pause between injection and back production. Since the ester and alcohol are starting at the same point in space, they will arrive at different times. This time difference can be used to calculate the residual oil saturation in the near well zone. The formulae used is:

$$S = \frac{(t_2 - t_1)}{(t_2 + t_1(K-1))} \tag{42}$$

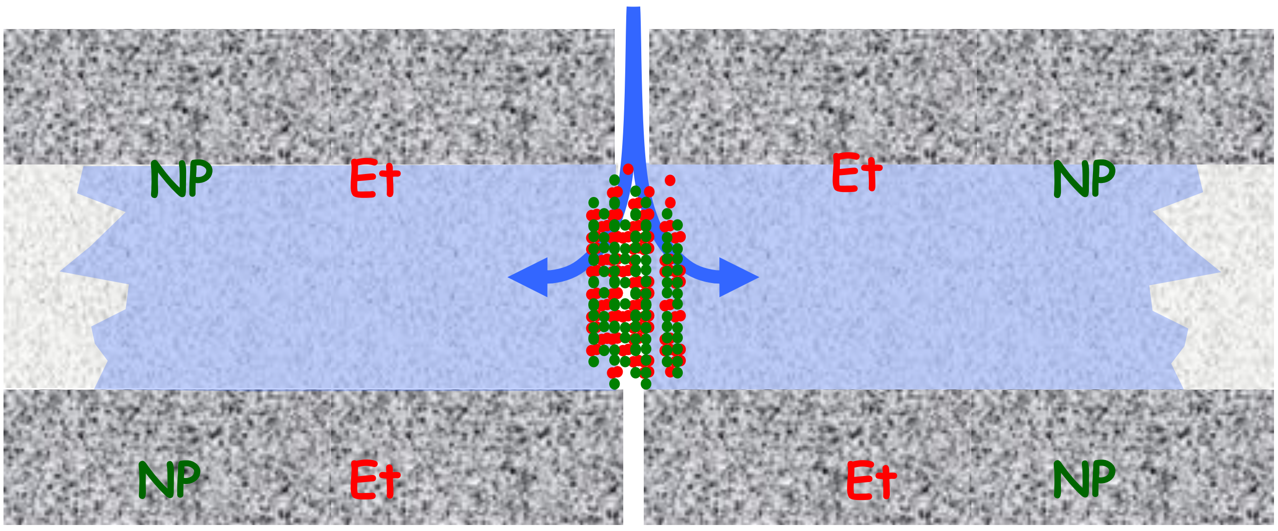


Figure 21. Schematic sketch of the SWCTT situation

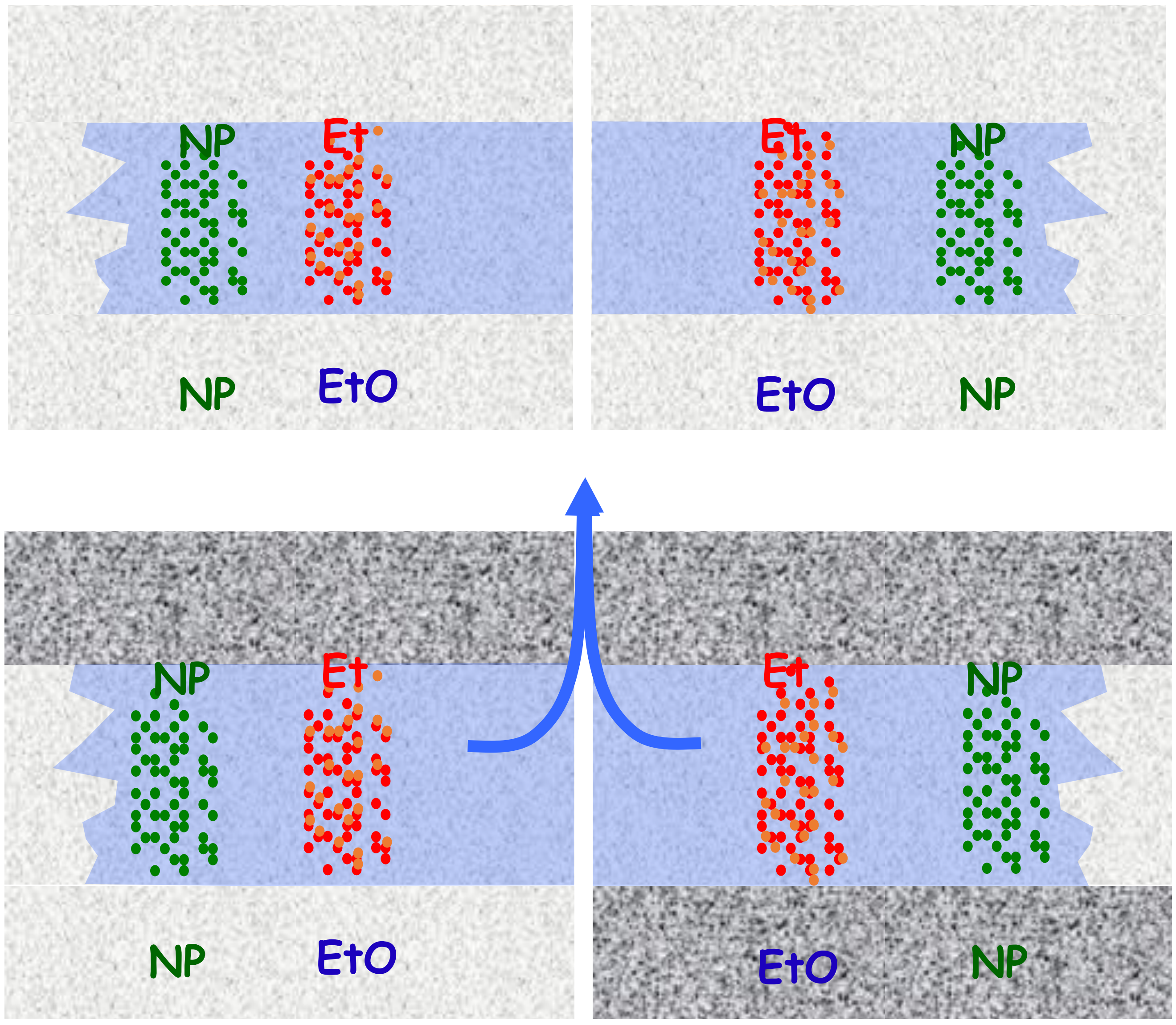


Figure 22. Illustrating the reaction period (no injection) and back production

The process of injection ester and back producing ester and alcohol, is illustrated in the figures on previous page.

The ester undergoes partitioning between water and oil, the saturations vary in time and space, the ester and alcohol components are subjected to varying phase velocities and are subjected to physical dispersion varying with the phase velocities. We can now write the transport-reaction equation for ester as

$$\frac{\partial}{\partial t}\left(\sum_{i=o,g,w}\varphi\, S_i\, K_i^e C^e\right)+\nabla\cdot\left(\sum_{i=o,g,w}\boldsymbol{v}_i K_i^e C^e\right)-\nabla\cdot\left(\sum_{i=o,g,w}\varphi S_i \boldsymbol{D}_i^*\cdot\nabla(K_i^e C^e)\right) = -\sum_{i=o,g,w}\varphi\, S_i\, K_i^e\, \kappa_i^e\, C^e \qquad (43)$$

where κ_i^e is the hydrolysis reaction rate in phase i. κ_i^e is zero for oil and gas and equals the hydrolysis rate of ester (κ) in water. For alcohol we can write

$$\frac{\partial}{\partial t}\left(\sum_{i=o,g,w} \varphi\, S_i\, K_i^a C^a\right) + \nabla\cdot\left(\sum_{i=o,g,w} \boldsymbol{v}_i K_i^a C^a\right) - \nabla\cdot\left(\sum_{i=o,g,w} \varphi S_i \boldsymbol{D}_i^* \cdot \nabla(K_i^a C^a)\right) = \sum_{i=o,g,w} \varphi\, S_i\, K_i^a\, \kappa_i^e\, C^e \frac{M_a}{M_e} \qquad (44)$$

Note that the 2-component system of equations in general would have to be solved simultaneously. In our case, however, the equations are only 1-way dependent. Hence we solve the ester equation first, and then the alcohol equation can be solved afterwards, once the ester concentration C^e has been determined.

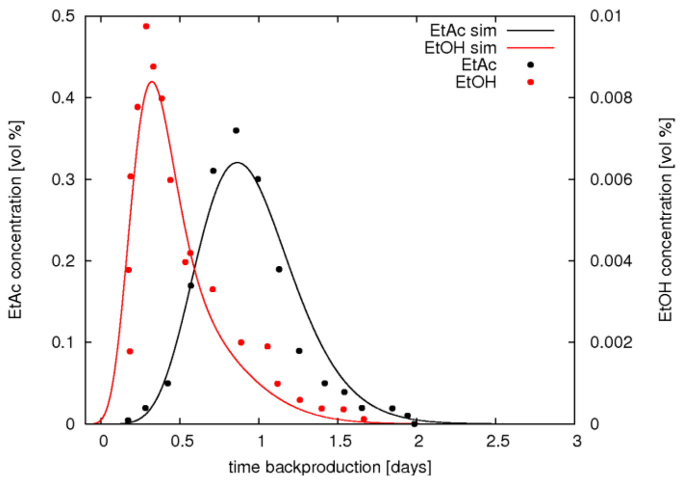


Figure 23. The back produced tracer signals (alcohol and ester)

The National IOR Centre of Norway

The discretized well model in IORSim

A production well in Eclipse can either be simulated in crossflow mode or not. If the crossflow option is turned on, the flow in vicinity of the well may look as in the figure below:

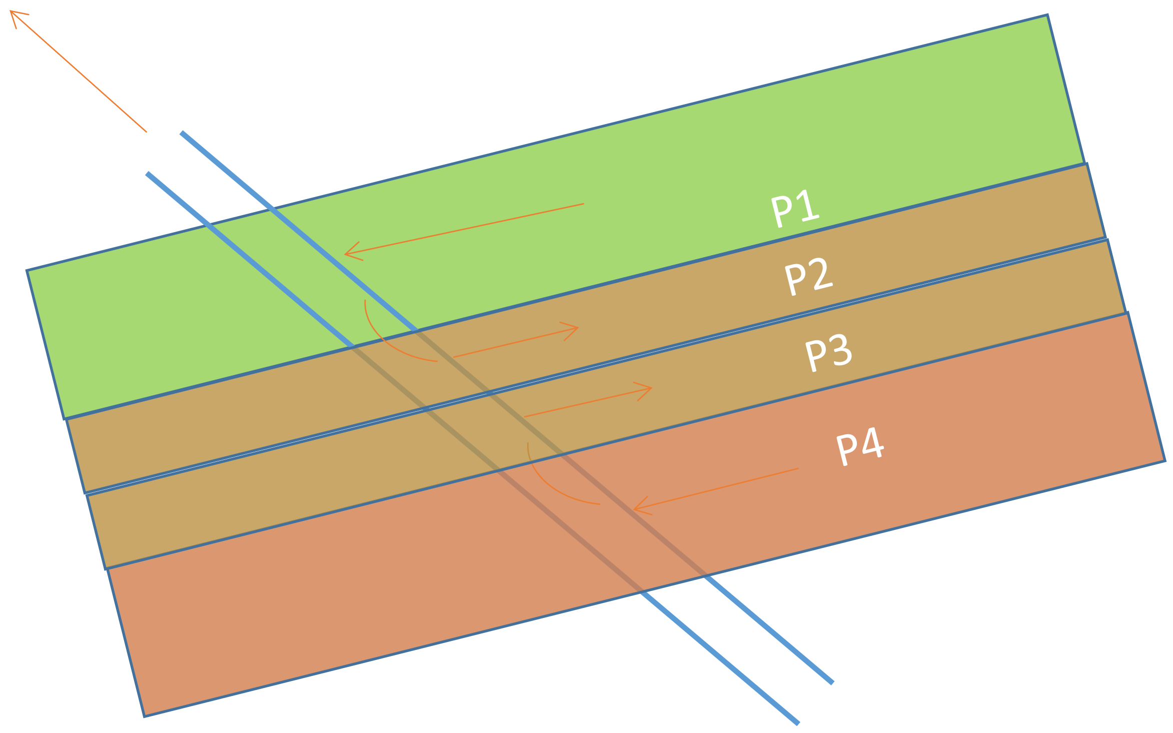


Figure 24. Crossflow in well, arising due to different reservoir layers

As shown in the figure, water, oil and gas may flow through the well from one reservoir layer to another. This implies that in some completion zones the flow will go out from the well, while in other zones the flow may go into the well, although the well is a producer which implies that the total production is positive. In IORSim there are also two options, either to include crossflow or not. If no crossflow is assumed, the well is treated quite simply, just calculating and summing together, the species production contribution from each layer, and calculating the total species production and the average species concentration in the production well.

In the case of crossflow, a more complicated well model is needed in IORSim. As seen in the figure below, the well is in this case discretized by imposing extra internal pipe cells along the well. These pipe cells are incorporated in the total grid system of IORSim, as any other grid cells. The flowrate for each pipe connection (boundary between pipe cells) are calculated by propagating from the toe of the well, adding together flow contributions for each completion zone. The result is that the varying internal flow between individual pipe cells is calculated and stored in grid cell connections, which are between two pipe cells or between the pipe cell and the adjacent reservoir layer holding the completion.

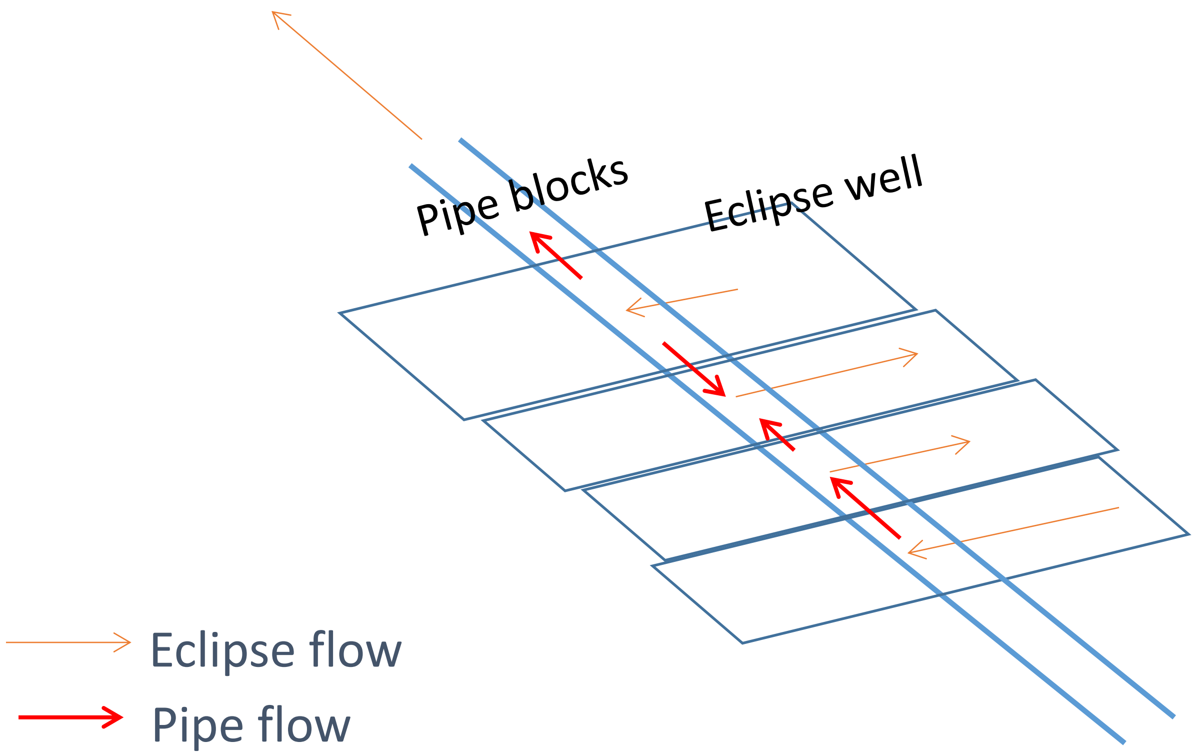


Figure 25. Crossflow in the well, the representation in the IORSim grid system

The result is that the transport of species can be simulated throughout the well, and that a total production rate and a produced concentration can be calculated for the well.

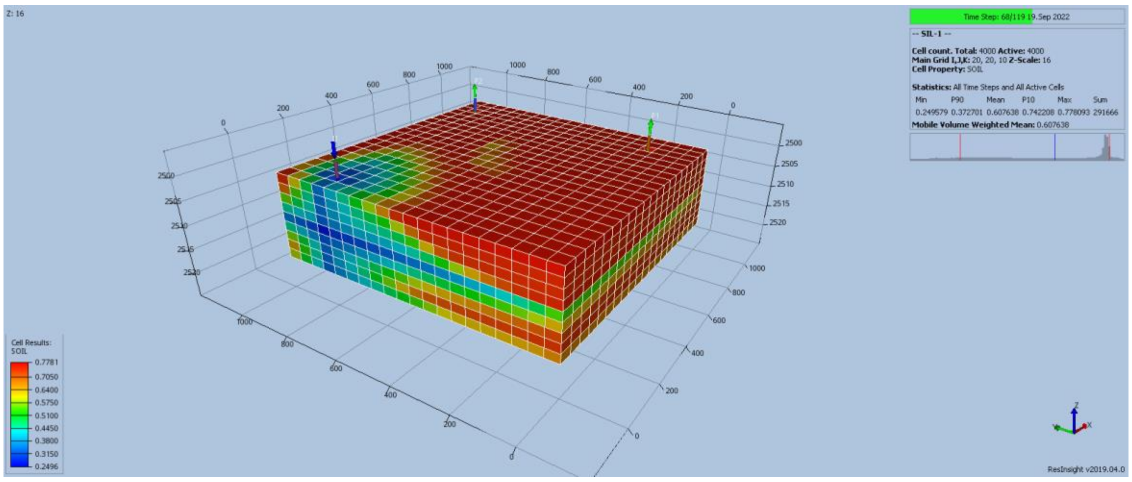
Examples

In this section we describe two field scale applications of IORSim for silicate and low salinity option.

Silicate model

This model, called SILICATE-1, with small modifications was used in the IORSim course of 10.12.2020. The model contains one injection well and two production wells in a reservoir with two high permeable zones. All model details can be found in the SILICATE-1 files in the IORSim directory.

The sodium silicate solution is injected for 30 days after 300 days of pure water flooding. A picture showing the oil saturation after slightly more than 1.5 years can be seen in fig. 26. The location of the immobile gel as well as silicate/gel concentration in two adjacent grid blocks, outside and inside the geled area is seen in fig. 27.



Time Step: 68/119 19.Sep 2022

-- SIL-1 --

Cell count. Total: 4000 **Active:** 4000
Main Grid I,J,K: 20, 20, 10 **Z-Scale:** 16
Cell Property: SOIL

Statistics: All Time Steps and All Active Cells

```
Min       P90       Mean      P10       Max       Sum
0.249579  0.372701  0.607638  0.742208  0.778093  291666
```

Mobile Volume Weighted Mean: 0.607638

Figure 26. Model with two high permeable layers used for silicate injection simulation

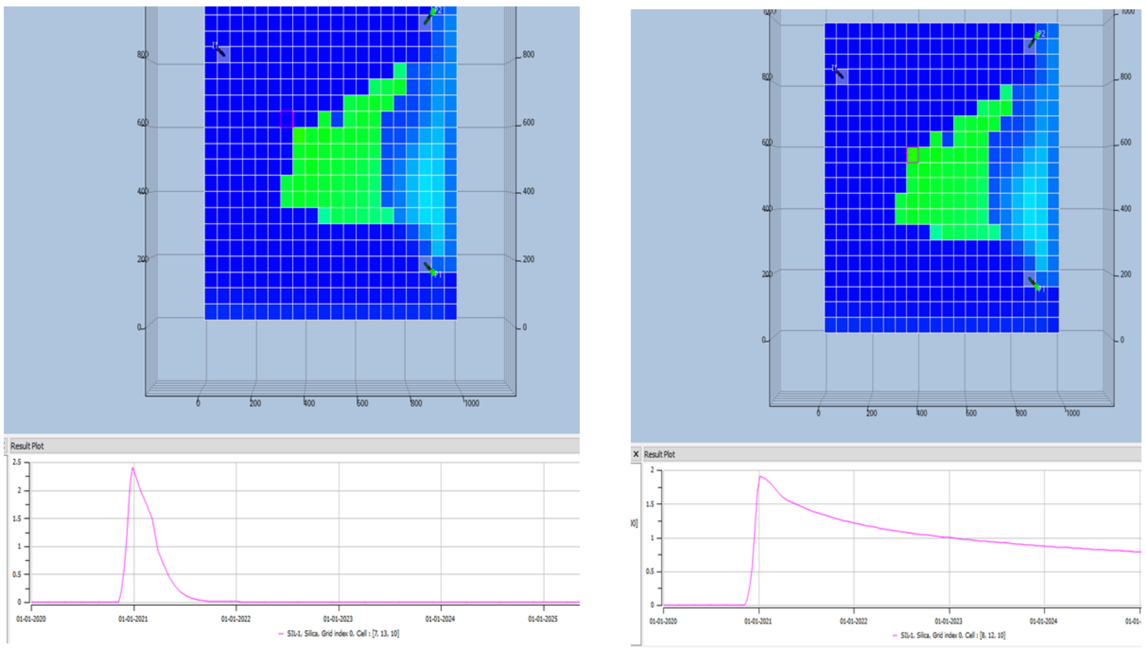


Figure 27. Silicate gel as seen from above in high permeable bottom layer. Silicate concentration history for two adjacent grid blocks, left; outside immobile gel, right; inside.

The simulated improved oil recovery for the silicate injection compared pure waterflood is shown in fig. 28. Cumulative oil production has increased by 90,000 Sm3 (7%) at end of simulation (2000d = 5.5y). Total water production was reduced by 320,000 Sm3 (17.5%).

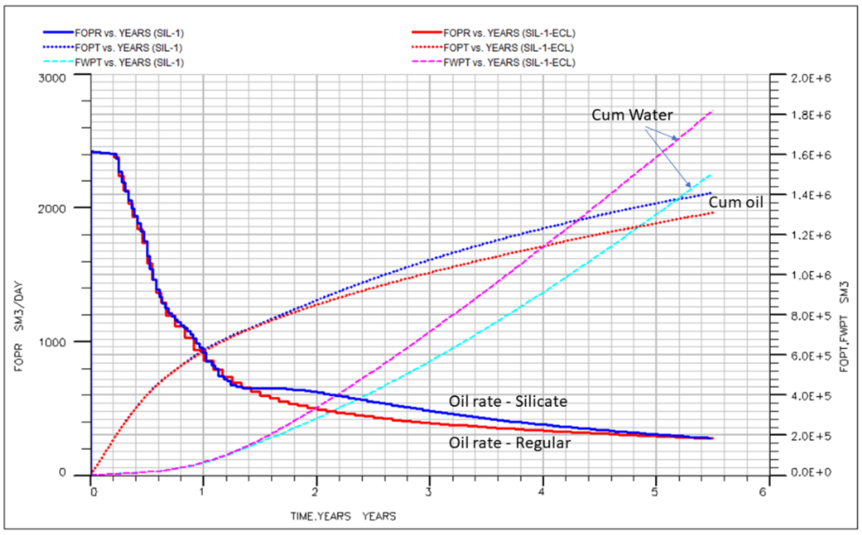


Figure 28. Silicate model oil recovery with blue curves compared with regular flood in red/pink curves.

Low salinity model

In the low salinity model included here adsorbed magnesium concentration is assumed to correlate with the reduction in residual oil. Alternatives in IORSim are any ionic species in water, or adsorbed, as well as pH.

The injection water contains a higher magnesium concentration than the formation water. As has been explained IORSim also includes a geochemical reactions module.

Data files including all details can be found in the LOWSAL-1 files in the IORSim directory.

Adsorbed magnesium concentration after 16y of waterflooding is illustrated in fig. 29.

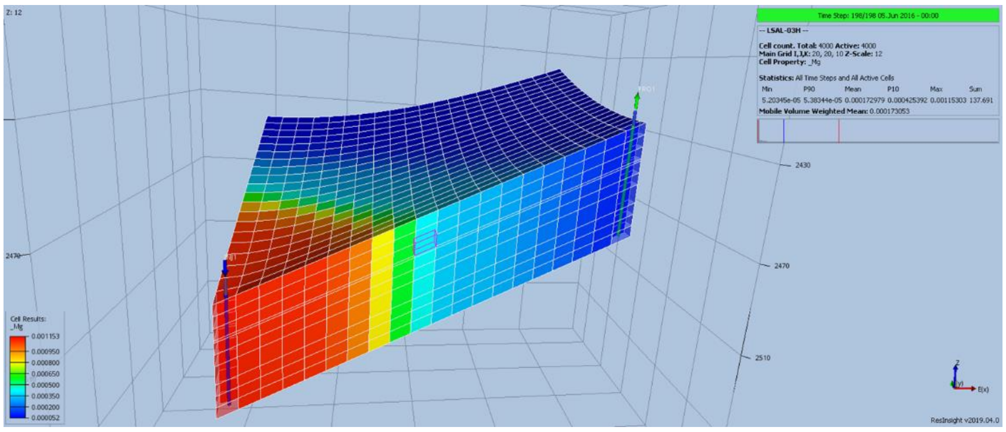

```
Time Step: 198/198 05.Jun 2016 - 00:00
-- LSAL-03H --
Cell count. Total: 4000 Active: 4000
Main Grid I,J,K: 20, 20, 10 Z-Scale: 12
Cell Property: _Mg

Statistics: All Time Steps and All Active Cells
 Min         P90         Mean        P10         Max         Sum
 5.20345e-05 5.38344e-05 0.000172979 0.000425392 0.00115303 137.691
Mobile Volume Weighted Mean: 0.000173053
```

Figure 29. Adsorbed magnesium concentration after 16 years of flooding

The corresponding oil saturation is shown in fig. 30.

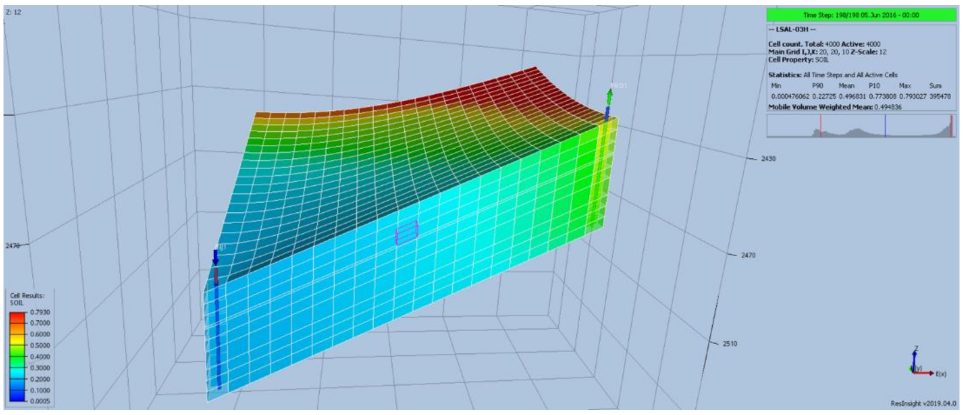

```
Time Step: 198/198 05.Jun 2016 - 00:00
-- LSAL-03H --
Cell count. Total: 4000 Active: 4000
Main Grid I,J,K: 20, 20, 10 Z-Scale: 12
Cell Property: SOIL

Statistics: All Time Steps and All Active Cells
 Min         P90     Mean     P10      Max      Sum
 0.000476062 0.22725 0.496831 0.773808 0.793027 395478
Mobile Volume Weighted Mean: 0.494836
```

Figure 30. Oil saturation for Low Salinity flood model

A comparison with regular waterflood is shown in fig. 31.

(In fig. 31 the low salinity case has the name LSAL-03H and the regular flood is named LSAL03E2-ECL.)

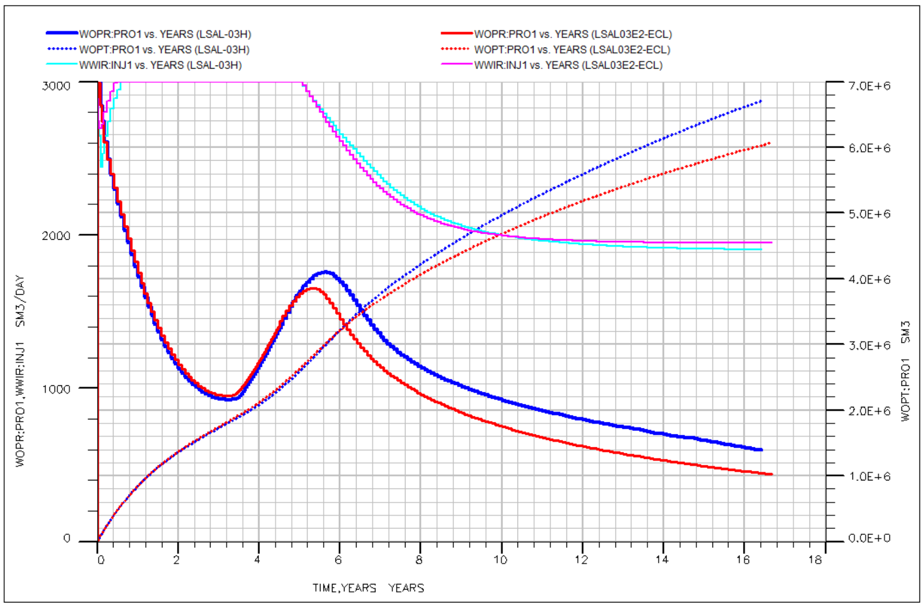


Figure 31. Low salinity flood with blue curves, regular flood with red/pink curves. WOPR is oil rate, WOPT is cumulative oil and WWIR is injection rate.

About 12% more oil was simulated being produced with low salinity flood in this model.

User guide for the IORSim script

The main motivation for IORSim was to add geochemistry based IOR methods to field simulation workflows without altering the reservoir simulator. To achieve this, the communication between IORSim and the reservoir simulator operate via interface-files. The current version of IORSim is developed to cooperate with Eclipse100. However, with some modifications, other reservoir simulators that allow communication via files can cooperate with IORSim.

IORSim is executed in forward or backward mode. In forward mode, the Eclipse run completes before IORSim reads the Eclipse output and perform the calculations. In backward mode, Eclipse only executes one timestep before it is paused. IORSim then reads the timestep output and prepare an interface-file for Eclipse with updated keywords (such as SATNUM). IORSim is then paused and Eclipse resume to read the updated interface-file.

A python script is developed to handle the tasks of pausing and resuming Eclipse and IORSim, checking that output files a properly flushed, and preparing interface-files. The script can be executed in graphics mode (GUI), on the command line (terminal), or imported and used by other python scripts. The capabilities of the script include

- Run IORSim in backward, forward, and single mode
- Convert and merge IORSim and Eclipse restart files to make the results viewable in ResInsight
- Check IORSim input keywords

- Progress-bar with estimated remaining simulation time

The GUI version of the script includes these additional features

- Plot Eclipse and IORSim well-data, with an option to compare two cases
- Edit Eclipse and IORSim input files
- View log files
- Integrated searchable IORSim user guide

A running version of Eclipse and IORSim is required by the script, and the script is compatible with Linux and Windows. The latest version of the script can be downloaded from github.com/janlv/IORSim_GUI/releases/latest.

The GUI window

The Run-, Case- and Days-fields of the toolbar are the input variables of the script. The Run-field is a dropdown-menu with four different running modes: Forward, Backward, Eclipse, and IORSim. The Case-field is a dropdown-menu of all imported cases, and the Days-field is the total simulation time. The green triangle starts the simulation, and the red square allow the user to stop the simulation before the total simulation time is reached. The Compare-field is a dropdown-menu similar to the Case-field that allows the user to compare plots from two different cases.

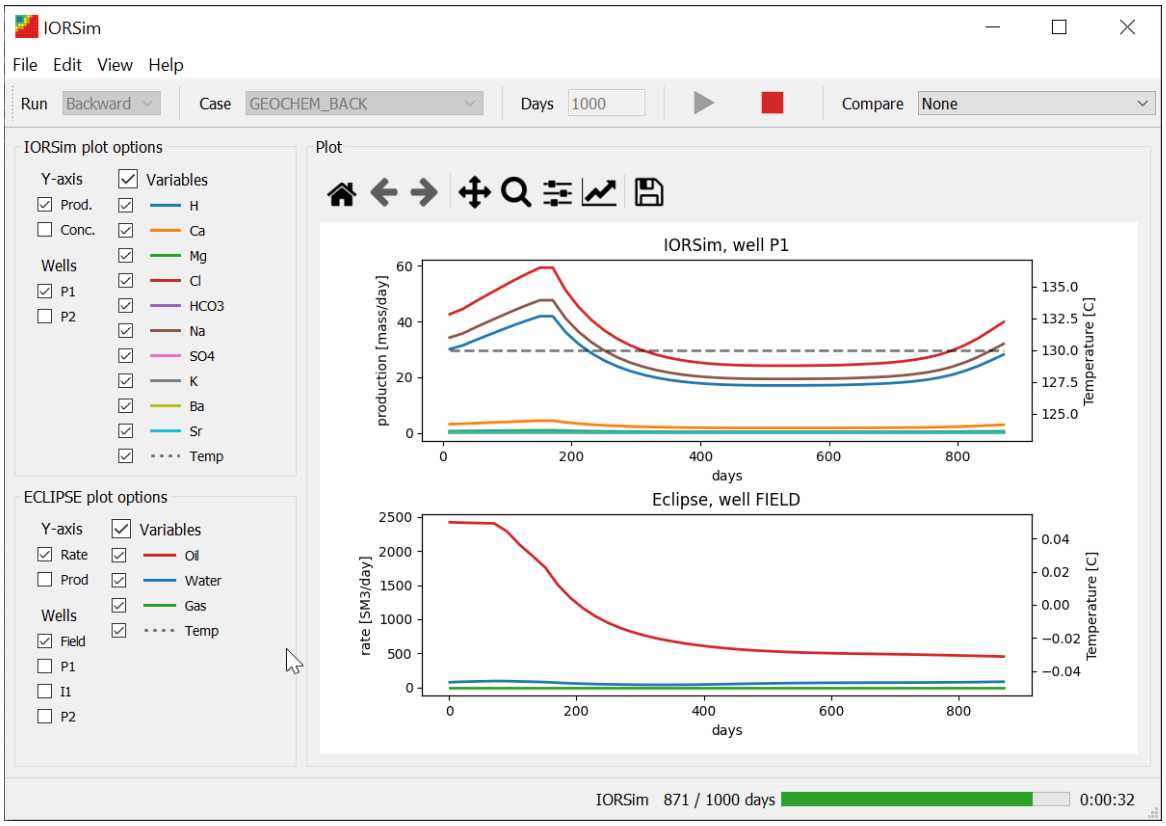
The main view of the GUI is the plot window and the Eclipse and IORSim plotting menus. This allows the user to follow the evolution of selected well-data during the simulation. A maximum of two plots can be view simultaneously. If more than two Y-axis boxes or more than two Wells boxes are selected, the last selected box is automatically unselected.

The Edit-menu opens an editor for Eclipse or IORSim input files with syntax highlighting and a search function. The View-menu opens a log file viewer showing the terminal output of Eclipse or IORSim runs, or the progress of the python script. The editor and log viewer are displayed in the plot window. The Plot option at the top of the View-menu brings back the default plot view.

The Edit-menu also gives access to a Settings window where the most important entry is the location of the executable IORSim program. This needs to be specified before running the first simulation. The remaining options are described in section 1.1.5.

The Help-menu opens a searchable IORSim user guide in a separate window

A status-bar showing the progress of the simulation and the estimated remaining time is located at the bottom of the GUI window.



Run Backward   Case GEOCHEM_BACK   Days 1000   Compare None

IORSim 871 / 1000 days 0:00:32

Figure 3: The graphical user interface (GUI) version of the IORSim script.

Running modes

Four different running modes are implemented and available from the Run dropdown-list: 1) Forward, 2) Backward, 3) Eclipse, and 4) IORSim

In Forward mode, Eclipse is first executed from start to finish to create restart (UNRST) and well-data (RFT) files. Then, IORSim reads the output from Eclipse and add tracer data, geochemical concentration values, or other relevant values in an Eclipse formatted restart file (FUNRST).

In Backward mode, Eclipse executes only one timestep before IORSim reads the Eclipse output and updates the flow variables. This allows IORSim to update the SATNUM values used by Eclipse in the next timestep. During a backward run, the Eclipse output files (UNRST and RFT) need to be checked to ensure they are properly flushed before Eclipse is paused and IORSim can resume.

The Eclipse and IORSim running modes simply execute Eclipse and IORSim as stand-alone applications. The IORSim mode can only be executed if Eclipse output files are present in the case-folder.

Case files

A minimal IORSim case consists of an Eclipse input file (.DATA-file) and an IORSim input file (.trcinp-file). If the geochemical option is used a chemistry-file must be included using the *CHEMFILE keyword in the IORSim input file. The Eclipse input file may also include different additional files. The case-files for a specific case are organized in separate case-folders under 'IORSim_cases' in the running directory. The path and name of the main case directory can be changed from the Settings window (Edit -> Settings).

For a DATA-file named Case.DATA the following files are copied to the folder CASE if they exist

- The Eclipse input file, Case.DATA
- The IORSim input file, Case.trcinp
- All files ending with .inc, .dat, .grdecl, .egrid, .vfp, or .sch
- All files included by the *CHEMFILE keyword in the trcinp-file

Note that the filename match is case-insensitive, also if executed under Linux.

Import and run an existing case

Import an existing case using File -> Import case… to bring up a file-navigation window where you can locate the .DATA-file of the case you want to import. After import, the name of the new case will appear in capital letters in the Case dropdown-menu in the toolbar. The running mode is selected from the Run dropdown-menu. A backward run is automatically selected if the READDATA keyword is present in the DATA-file. A Forward run is the default for all other cases. For forward-cases, Eclipse and IORSim can also be run in single mode by choosing Eclipse or IORSim from the Run menu.

For a backward case, the duration of the simulation is given in the Days field of the toolbar. For a forward-case, the total simulation time, given by the TSTEP keyword in the DATA-file, is displayed in the Days field without the option to edit it directly. However, the Eclipse input file can be opened for editing via the Edit menu, and the simulation time modified by searching for TSTEP and changing the value.

The first time the script is used you need to give the location of the IORSim executable under Edit -> Settings. Clicking the green go-button will also bring up the Settings-window shown in Figure 4 if the IORSim program is missing. Locate the IORSim executable using the Open-button.

If Eclipse is properly installed, Eclipse can be executed by the default eclrun macro. If this should fail, the full path of the eclrun macro must be given in Settings. The remaining Setting-options are discussed later.

Start the simulation by clicking the green triangle in the toolbar. The progress-bar at the bottom shows the estimated remaining simulation time and displays the current timestep of the simulation.

By default, the view area shows evolving plots of the well-data from both Eclipse and IORSim output-files. Which values to plot are selected in the IORSim and Eclipse plotting menus to the left of the view area. The View menu also gives access to log-files showing the printed output from Eclipse, IORSim or the python script.

Settings

The settings window is available at the bottom of the Edit-menu, or via the shortcut Ctrl+s.

IORSim program

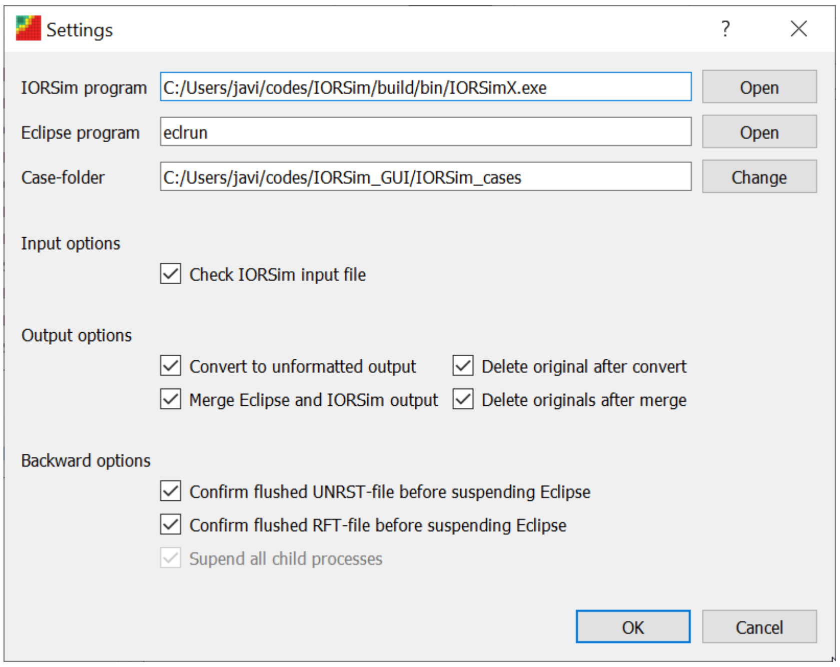
The location of the executable IORSim program is the only setting that require input from the user before running the first simulation.

Eclipse program

The default value ‘eclrun’ require that the Path environment variable includes the full path of the ecl/macros-folder. Otherwise, the full path of the eclrun executable must be specified using the Open button.

Input options

If checked, a simple check of the keywords in the IORSim input file is performed before the simulation starts. This will give a more useful feedback than if the IORSim application stops due to missing keywords or keywords in the wrong order.



IORSim program: C:/Users/javi/codes/IORSim/build/bin/IORSimX.exe

Eclipse program: eclrun

Case-folder: C:/Users/javi/codes/IORSim_GUI/IORSim_cases

Input options

- ✓ Check IORSim input file

Output options

- ✓ Convert to unformatted output
- ✓ Delete original after convert
- ✓ Merge Eclipse and IORSim output
- ✓ Delete originals after merge

Backward options

- ✓ Confirm flushed UNRST-file before suspending Eclipse
- ✓ Confirm flushed RFT-file before suspending Eclipse
- ✓ Supend all child processes

Figure 4: The settings window of the IORSim GUI script

Output options

IORSim outputs reservoir data in an ASCII restart file (with the extension FUNRST) that follow the Eclipse format. The data can be viewed in the FloViz program that is part of the Eclipse installation. However, some prefer to use ResInsight for reservoir visualization, but ResInsight can only import binary (unformatted) Eclipse restart files. In addition, Eclipse and IORSim data are saved in separate files which makes it a bit cumbersome to get all the reservoir data available in the same view.

By checking the convert and merge boxes, the script first converts the ASCII output to binary output and then merge the original Eclipse and converted IORSim restart file into one unified restart file. By default, the original files are deleted after a successful convert and merge to save disk space. Uncheck the two delete-boxes to keep the original restart files.

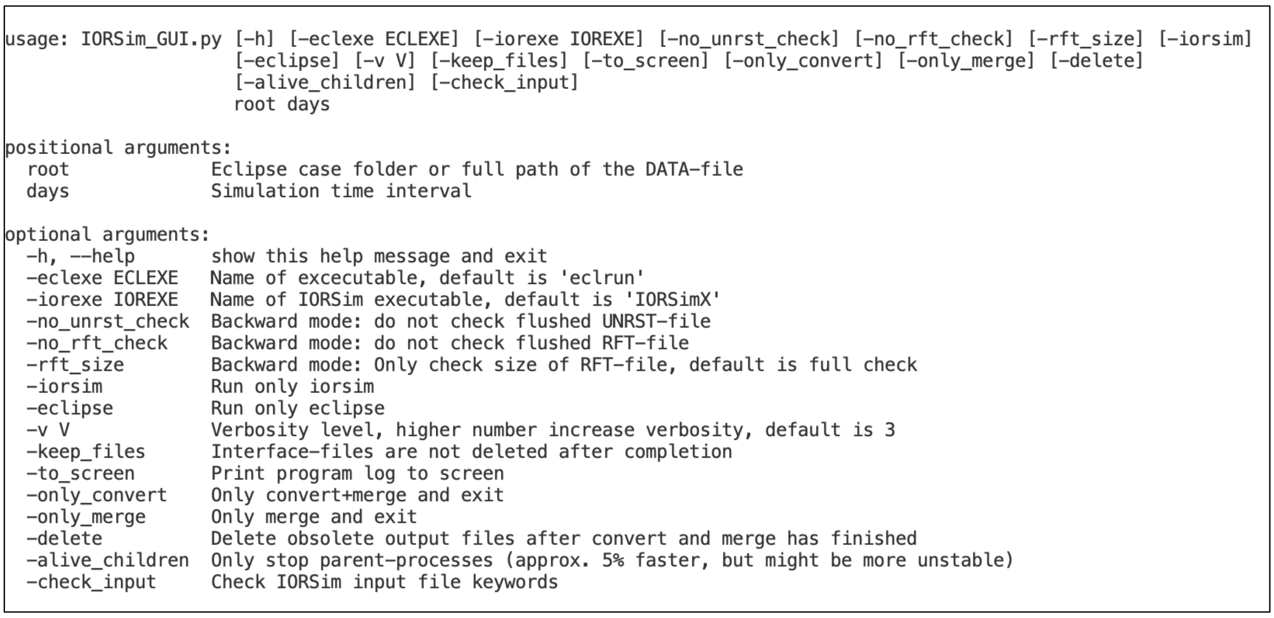
Backward options

During a backward run it is important to check that the Eclipse restart file (the UNRST file) and the well-data file (the RFT file) are completely flushed before Eclipse is paused. Disabling these checks might speed up the simulation, but it will make backward runs less stable and prone to stop prematurely.

The option 'Suspend all child processes' means that all sub-processes of a running processes are suspended when the main process is suspended. This option is always on and unable to uncheck since it improves the stability of backward runs. However, the option can be disabled by giving the -alive_children argument to the command line version of the script (see Section 0)

Using the script on the command line (non-GUI version)

The script can also be executed on the command line by providing a path to a DATA-file and the simulation duration in days. The different arguments are listed and explained in Figure 5. It is recommended to run the GUI version of the script first and locate the IORSim executable in the Settings window to avoid specifying the executable on the command line every time the script is used.

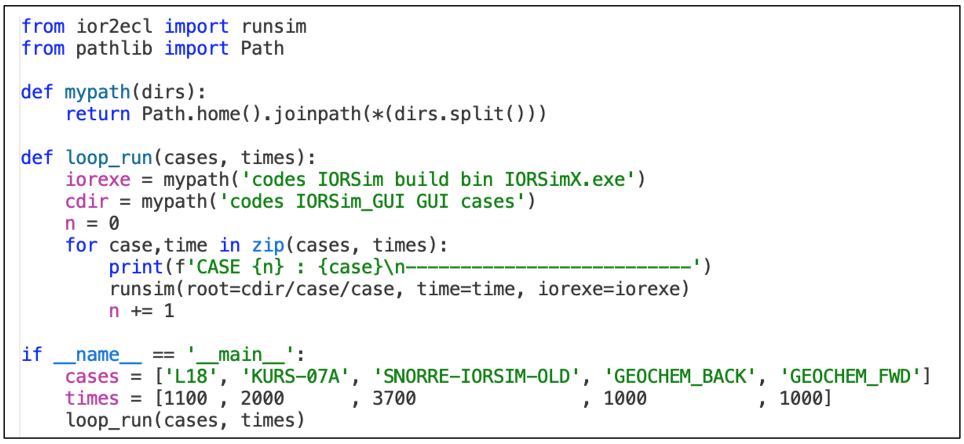


```
usage: IORSim_GUI.py [-h] [-eclexe ECLEXE] [-iorexe IOREXE] [-no_unrst_check] [-no_rft_check] [-rft_size] [-iorsim]
                     [-eclipse] [-v V] [-keep_files] [-to_screen] [-only_convert] [-only_merge] [-delete]
                     [-alive_children] [-check_input]
                     root days

positional arguments:
  root               Eclipse case folder or full path of the DATA-file
  days               Simulation time interval

optional arguments:
  -h, --help         show this help message and exit
  -eclexe ECLEXE     Name of excecutable, default is 'eclrun'
  -iorexe IOREXE     Name of IORSim executable, default is 'IORSimX'
  -no_unrst_check    Backward mode: do not check flushed UNRST-file
  -no_rft_check      Backward mode: do not check flushed RFT-file
  -rft_size          Backward mode: Only check size of RFT-file, default is full check
  -iorsim            Run only iorsim
  -eclipse           Run only eclipse
  -v V               Verbosity level, higher number increase verbosity, default is 3
  -keep_files        Interface-files are not deleted after completion
  -to_screen         Print program log to screen
  -only_convert      Only convert+merge and exit
  -only_merge        Only merge and exit
  -delete            Delete obsolete output files after convert and merge has finished
  -alive_children    Only stop parent-processes (approx. 5% faster, but might be more unstable)
  -check_input       Check IORSim input file keywords
```

Figure 5 Description of the command line arguments of the IORSim python script. This information is displayed if -h is given as an argument to the script

Using the script as a python module

The script can also be imported as a python module and used in other python scripts or Jupyter notebooks. Figure 6 shows a python script that use the IORSim script to loop over a



```python
from ior2ecl import runsim
from pathlib import Path

def mypath(dirs):
    return Path.home().joinpath(*(dirs.split()))

def loop_run(cases, times):
    iorexe = mypath('codes IORSim build bin IORSimX.exe')
    cdir = mypath('codes IORSim_GUI GUI cases')
    n = 0
    for case,time in zip(cases, times):
        print(f'CASE {n} : {case}\n--------------------------')
        runsim(root=cdir/case/case, time=time, iorexe=iorexe)
        n += 1

if __name__ == '__main__':
    cases = ['L18', 'KURS-07A', 'SNORRE-IORSIM-OLD', 'GEOCHEM_BACK', 'GEOCHEM_FWD']
    times = [1100 , 2000      , 3700               , 1000          , 1000]
    loop_run(cases, times)
```

Figure 6: Possible python implementation that use the IORSim script to loop over a list of case-folders

==

(i) Backward coupling
(ii) GUI
(iii) Examples

i) Conclusion(s)

The basic concept in IORSim coupling a commercial simulator with a research simulator dynamically works well. Thereby the application of common reservoir simulators can be extended to IOR processes originally missing in commercial software.

ii) Future work/plans
(Work in progress, journal papers in particular)

Felix publications

iii) Dissemination of results
(Include testing/implementation by research-/user partners if relevant)

Presentation IOR Conference 2015 “IORSim – an add on tool to ECLIPSE for fast and accurate simulation of multi-phase geochemical interactions at the field scale” Video presentation”

IOR Conference 2016 publication “IORSim an add on tool to ECLIPSE for simulating IOR processes” A. Hiorth, J. Sagen, A. Lohne, J. Nossen, J.L. Vinningland, E. Jettestuen and T. Sira,

“IORSim - A Simulator for Fast and Accurate Simulation of Multi- phase Geochemical Interactions at the Field Scale”, ECMOR XV - 15th European Conference on the Mathematics of Oil Recovery, 10.3997/2214-4609.201601882 Hiorth, Aksel; Sagen, Jan; Lohne, Arild; Nossen, Jan; Omekeh, Aruoture Voke; Stavland, Arne; Haukås, Jarle; Sira, Terje.

Simulating gelation of silica for in-depth reservoir plugging using IORSim as an add on tool to ECLIPSE. PEA EOR Forum; 2016-09-14 - 2016-09-14 Hiorth, Aksel; Sagen, Jan; Lohne, Arild; Nossen, Jan; Omekeh, Aruoture Voke; Stavland, Arne; Haukås, Jarle; Sira, Terje.

Simulating gelation of silica for in-depth reservoir plugging using IORSim as an add on tool to ECLIPSE. IEA EOR 2016; 2016-09-19 - 2016-09-21 A. Hiorth, J. Sagen, A. Lohne, A. Omekeh, J. Nossen, J. Haukås and T. Sira.

Simulation of Sodium Silicate Water Diversion Using IORSim, IOR 2017 - 19th European Symposium on Improved Oil Recovery A. Hiorth

Hiorth, Aksel; Nødland, Oddbjørn Mathias; Stavland, Arne; Jettestuen, Espen; Aursjø, Olav; Vinningland, Jan Ludvig; Nossen, Jan; Sagen, Jan; Sira, Terje: Simulation Tools for Predicting IOR Potential on the Norwegian Continental Shelf. IOR NORWAY 2018, April 2018.

IORSim - Modelling of advanced IOR processes coupled to commercial simulators, VIII November Conference - Improved Oil Recovery Session, 2020, Brazil, B. Antonsen

IORSim – adding more physics and chemistry to reservoir simulators, IOR Norway 2021, A. Hiorth

IORSim - modelling of advanced IOR processes coupled to commercial simulators. IOR Session 25th November in November Virtual Conference 2020 PUC-Rio, A. Hiorth, J.L. Vinningland, J. Sagen, J. Nossen, T. Sira, S. Groland, B. Antonsen

IORSim – a simulation tool for improving IOR processes by adding geochemistry with fast tracer tracking to industrial reservoir simulators, SPE Virtual Workshop: Tracer Technology - Improving Asset Value, March 2021, J. K. Sveen (IFE)

IORSim – Adding more physics and chemistry to reservoir simulators. IOR Norway 2021, UIS 26-28 April, A. Hiorth, J. sagen, J. Nossen, A. Lohne, J. L, Vinningland, B. Antonsen, E. Brendsdal, T. Sira

IORSim – version 1.0 (2021)

References

Parkhurst, D.L. and Appello (1999). “Users’s guide to PHREEQC (Version 2): A computer program for speciation, batch-reaction, one-dimensional transport, and geochemical calculations”.

Aagaard, P. and H. Helgeson (1982). "Thermodynamic and kinetic constraints on reaction rates among minerals and aqueous solutions; I, Theoretical considerations." Am. J. Sci. **282**: 237-285.

Bethke, C. M. (1996). Geochemical Reaction Modeling. New York, Oxford University Press.

Borkovec, M. and J. Westall (1983). "Solution of the Poisson-Boltzmann equation for surface excesses of ions in the diffuse layer at the oxide-electrolyte interface." Journal of Electroanalytical Chemistry and Interfacial Electrochemistry **150**(1-2): 325-337.

Helgeson, H. and D. Kirkham (1974). "Theoretical prediction of the thermodynamic behavior of aqueous electrolytes at high pressures and temperatures; I, Summary of the thermodynamic/electrostatic properties of the solvent." Am. J. Sci. **274**: 1089-1198.

Helgeson, H. and D. Kirkham (1974). "Theoretical prediction of the thermodynamic behavior of aqueous electrolytes at high pressures and temperatures; II, Debye-Huckel parameters for activity coefficients and relative partial molal properties." Am. J. Sci. **274**: 1089-1198.

Helgeson, H., D. Kirkham and G. Flowers (1981). "Theoretical prediction of the thermodynamic behavior of aqueous electrolytes by high pressures and temperatures; IV, Calculation of activity coefficients, osmotic coefficients, and apparent molal and standard and relative partial molal properties to 600 degrees C and 5kb." Am. J. Sci. **281**: 1249-1516.

Hiorth, A., L. M. Cathles and M. V. Madland (2010). "The Impact of Pore Water Chemistry on Carbonate Surface Charge and Oil Wettability." Transport in Porous Media **85**(1): 1-21.

Hiorth, A., J. Sagen, A. Lohne, J. Nossen, A. V. Omekeh, A. Stavland, J. Haukås and T. Sira (2016). IORSim - an add on tool to ECLIPSE for simulating IOR processes. Sodium Silicate gelation and reservoir flow modification. IOR NORWAY 2016. Stavanger.

Israelachivili, J. (1985). Intermolecular and Surface Forces. New York City, Academic Press.

Johnson, J. W., E. H. Oelkers and H. C. Helgeson (1992). "SUPCRT92: A software package for calculating the standard molal thermodynamic properties of minerals, gases, aqueous species, and reactions from 1 to 5000 bar and 0 to 1000 C." Comp. and Geo. Sci. **18**(7): 899-947.

Minde, M. W. and A. Hiorth (2020). "Compaction and Fluid—Rock Interaction in Chalk Insight from Modelling and Data at Pore-, Core-, and Field-Scale." Geosciences **10 (1)**(6): 16.

Parkhurst, D. L. and C. Appelo (1999). "User's guide to PHREEQC (Version 2): A computer program for speciation, batch-reaction, one-dimensional transport, and inverse geochemical calculations."

Revil, A., P. A. Pezard and P. W. J. Glover (1999). "Streaming potential in porous media 1. Theory of the zeta potential." Journal of Geophysical Research-Solid Earth **104**(B9): 20021-20031.

Revil, A., P. A. Pezard and P. W. J. Glover (1999). "Streaming potential in porous media 1. Theory of the zeta potential." J. Geophys. Res. **104**(B9): 21-31.

Revil, A., H. Schwaeger, L. M. Cathles and P. D. Manhardt (1999). "Streaming potential in porous media 2. Theory and application to geothermal systems." Journal of Geophysical Research-Solid Earth **104**(B9): 20033-20048.