A global reactive transport model applied to the MoMaS benchmark

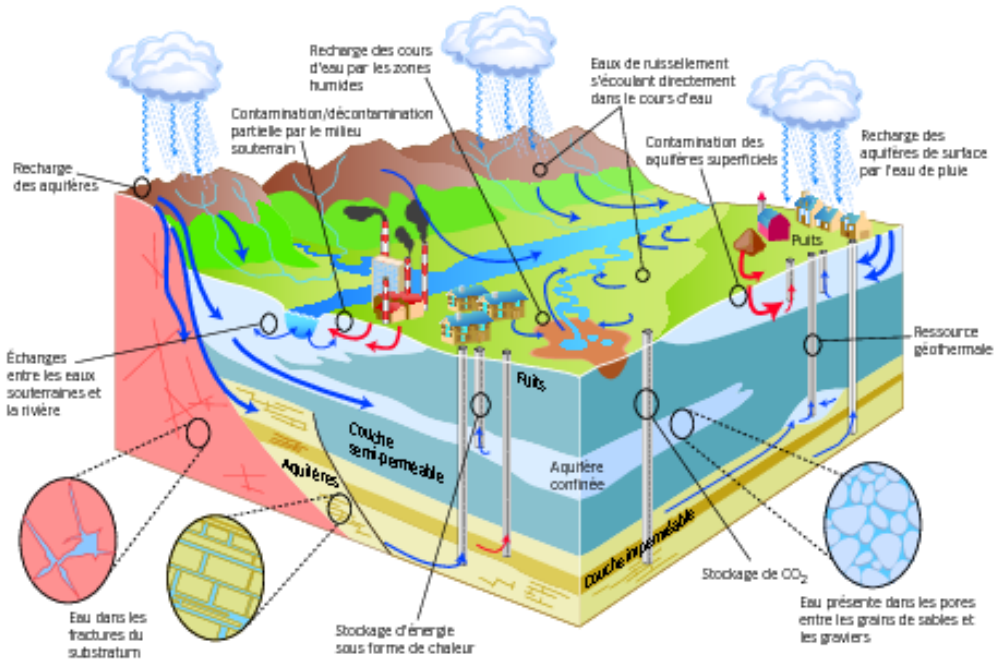
Jocelyne Erhel
SAGE team, Inria, RENNES

co-authors
Souhila Sabit (SAGE team, Inria, Rennes, France)

Pau, June 2015

1 Introduction

Water and energy resources



1062 tdc forage

- Manage water resources
- Prevent pollution
- Store waste, store energy, capture CO_2
- Use geothermal energy
- ...

Reactive transport

Flow model

- Single phase saturated flow
- Constant density and porosity. Heterogeneous conductivity
- Darcy's law and mass conservation law

Reactive transport

Flow model

- Single phase saturated flow
- Constant density and porosity. Heterogeneous conductivity
- Darcy's law and mass conservation law

Transport model

- Advection diffusion reaction
- Molecular diffusion and hydrodynamic dispersion
- Linear equations

Reactive transport

Flow model

- Single phase saturated flow
- Constant density and porosity. Heterogeneous conductivity
- Darcy's law and mass conservation law

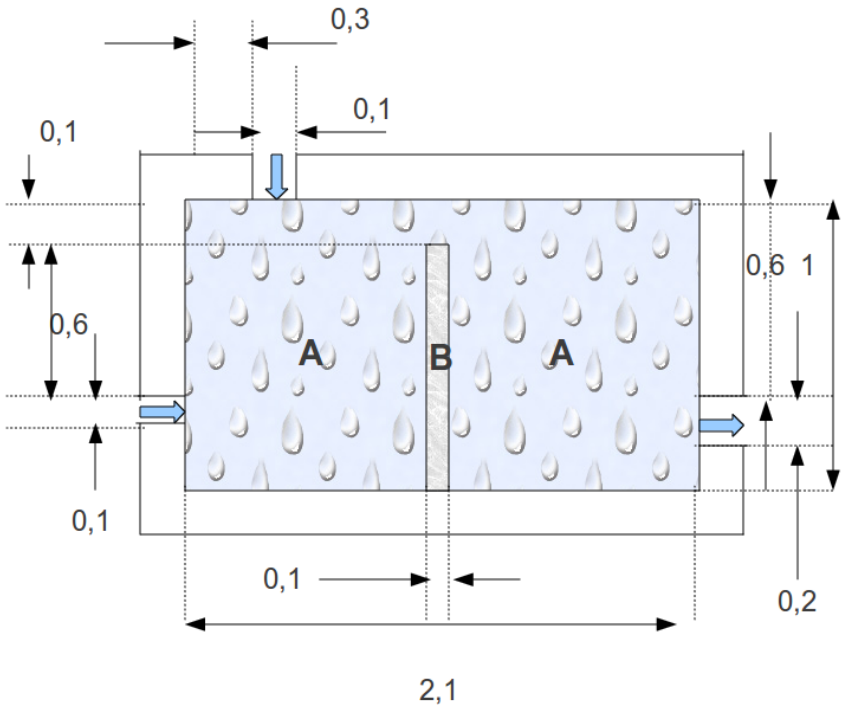
Transport model

- Advection diffusion reaction
- Molecular diffusion and hydrodynamic dispersion
- Linear equations

Reactive transport model

- Chemical equilibrium: components and secondary species
- Mass action laws and mass conservation law
- Transport of total analytical components

MoMaS benchmark: geometry and porous media



Physical parameters

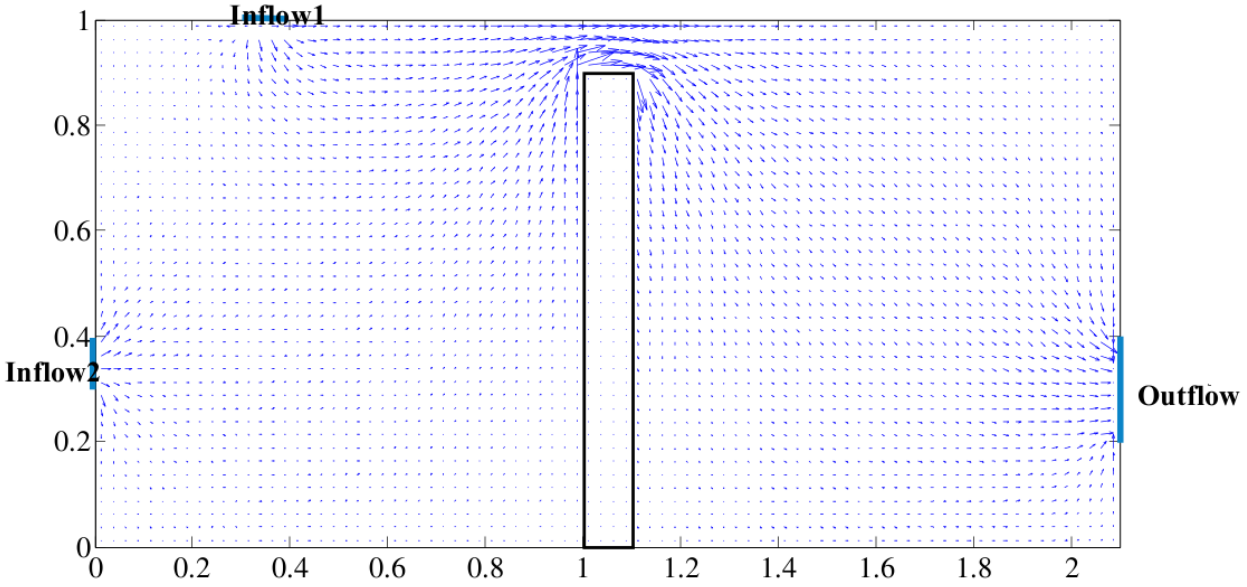
	Medium A	Medium B
Porosity ε	0.25	0.5
Conductivity K (L.T^{-1})	10^{-2}	10^{-5}

MoMaS benchmark: flow model

$$\begin{cases} q = -\mathbf{K}\nabla h, \\ \nabla q = 0. \end{cases}$$

Boundary conditions

Inflow	Outflow	Other
Velocity $q = 2.2510^{-2} L.T^{-1}$	Pressure $h = 1L$	No flow



MoMaS benchmark: transport model

$$\varepsilon \frac{\partial c}{\partial t} = \nabla.(D\ \nabla c) - \nabla.(q\ c)$$

$$D = \varepsilon d_m I + \alpha_T \|q\| I + (\alpha_L - \alpha_T) \frac{q q^T}{\|q\|}$$

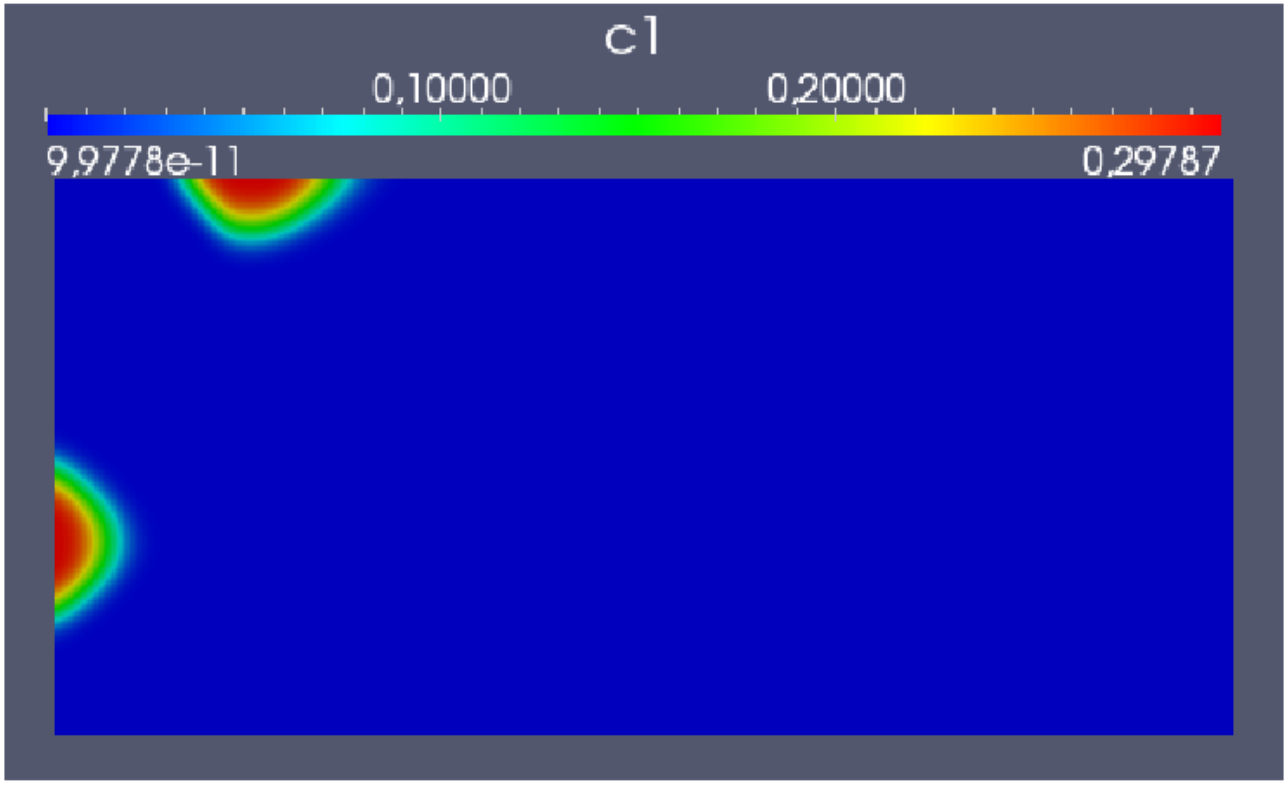
Physical parameters

	Medium A	Medium B
Molecular diffusion d_m $(L^2 T^{-1})$	0	0
Longitudinal Dispersion $\alpha_L(L)$	10^{-2}	6×10^{-2}
Transverse Dispersion $\alpha_T(L)$	10^{-3}	6×10^{-3}
Initial conditions	$c = 0$	$c = 0$

Boundary conditions

Inflow	Outflow	Other
Injection $0 \leq t \leq 5000\,T$: $c = 0.3$	No flux	No total flux
Leaching $5000\,T \leq t \leq 6000\,T$: $c = 0$	No flux	No total flux

MoMaS benchmark: simulation of an inert solute



Computations done using MT3D with a mesh of 80×168 cells

MoMaS benchmark: chemistry model

Chemical equilibrium

- Four primary aqueous components c_j and one primary fixed component s
- Five secondary aqueous species α_i and two secondary fixed species β_j

Stoichiometric coefficients and equilibrium constants

	c_1	c_2	c_3	c_4	s	K
α_1	0	-1	0	0	0	10^{-12}
α_2	0	1	1	0	0	1
α_3	0	-1	0	1	0	1
α_4	0	-4	1	3	0	0.1
α_5	0	4	3	1	0	$\mathbf{10^{+6}}$
β_1	0	3	1	0	1	10^{+6}
β_2	0	-3	0	1	2	10^{-1}

Chemical laws

Algebraic view of stoichiometric coefficients

	c	s	K
α	S	0	K_c
β	A	B	K_s

Chemical laws

Algebraic view of stoichiometric coefficients

	c	s	K
α	S	0	K_c
β	A	B	K_s

Mass action laws

$$\begin{cases} \alpha_i(c) = K_{ci} \prod_{j=1}^{N_c} c_j^{S_{ij}}, \quad i = 1, \dots N_\alpha, \\ \beta_i(c,s) = K_{si} \prod_{j=1}^{N_c} c_j^{A_{ij}} \prod_{j=1}^{N_s} s_j^{B_{ij}}, \quad i = 1, \dots N_\beta, \\ c_j \geq 0, \quad j = 1, \dots N_c, \\ s_j \geq 0, \quad j = 1, \dots N_s. \end{cases}$$

Chemical laws

Algebraic view of stoichiometric coefficients

	c	s	K
α	S	0	K_c
β	A	B	K_s

Mass action laws

$$\begin{cases} \alpha_i(c) = K_{ci} \prod_{j=1}^{N_c} c_j^{S_{ij}}, \quad i = 1, \dots N_\alpha, \\ \beta_i(c,s) = K_{si} \prod_{j=1}^{N_c} c_j^{A_{ij}} \prod_{j=1}^{N_s} s_j^{B_{ij}}, \quad i = 1, \dots N_\beta, \\ c_j \geq 0, \quad j = 1, \dots N_c, \\ s_j \geq 0, \quad j = 1, \dots N_s. \end{cases}$$

Mass conservation laws

$$\begin{cases} c + S^T \alpha(c) + A^T \beta(c,s) = T, \\ s + B^T \beta(c,s) = W, \end{cases}$$

Remarks about MoMaS geochemistry

Stoichiometric coefficients and equilibrium constants

	c_1	c_2	c_3	c_4	s	K
α_1	0	-1	0	0	0	10^{-12}
α_2	0	1	1	0	0	1
α_3	0	-1	0	1	0	1
α_4	0	-4	1	3	0	0.1
α_5	0	4	3	1	0	$\mathbf{10^{+6}}$
β_1	0	3	1	0	1	10^{+6}
β_2	0	-3	0	1	2	10^{-1}

Remarks

- $c_1 = T_1$ thus c_1 is an inert component
- $\alpha_1 = \frac{K_{c1}}{c_2}$ thus we assume that $c_2 > 0$
- If $T_3 = 0$ then $c_3 = \alpha_2 = \alpha_4 = \alpha_5 = \beta_1 = 0$
- If $T_4 = 0$ then $c_4 = \alpha_3 = \alpha_4 = \alpha_5 = \beta_2 = 0$
- If $W = 0$ then $s = \beta_1 = \beta_2 = 0$

Total analytical concentrations

Transport equation for each total analytical concentration

$$\varepsilon \frac{\partial T_j}{\partial t} = \nabla.(D\ \nabla C_j(c)) - \nabla.(q\ C_j(c)),\ j = 1, \ldots N_c$$

with the mobile part $C(c) = c + S^T \alpha(c)$

Total analytical concentrations

Transport equation for each total analytical concentration

$$\varepsilon \frac{\partial T_j}{\partial t} = \nabla.(D\ \nabla C_j(c)) - \nabla.(q\ C_j(c)),\ j = 1, \ldots N_c$$

with the mobile part $C(c) = c + S^T \alpha(c)$

Initial conditions

	T_1	T_2	T_3	T_4	W
Medium A	0	-2	0	2	1
Medium B	0	-2	0	2	10

Total analytical concentrations

Transport equation for each total analytical concentration

$$\varepsilon \frac{\partial T_j}{\partial t} = \nabla.(D\ \nabla C_j(c)) - \nabla.(q\ C_j(c)),\ j = 1, \ldots N_c$$

with the mobile part $C(c) = c + S^T \alpha(c)$

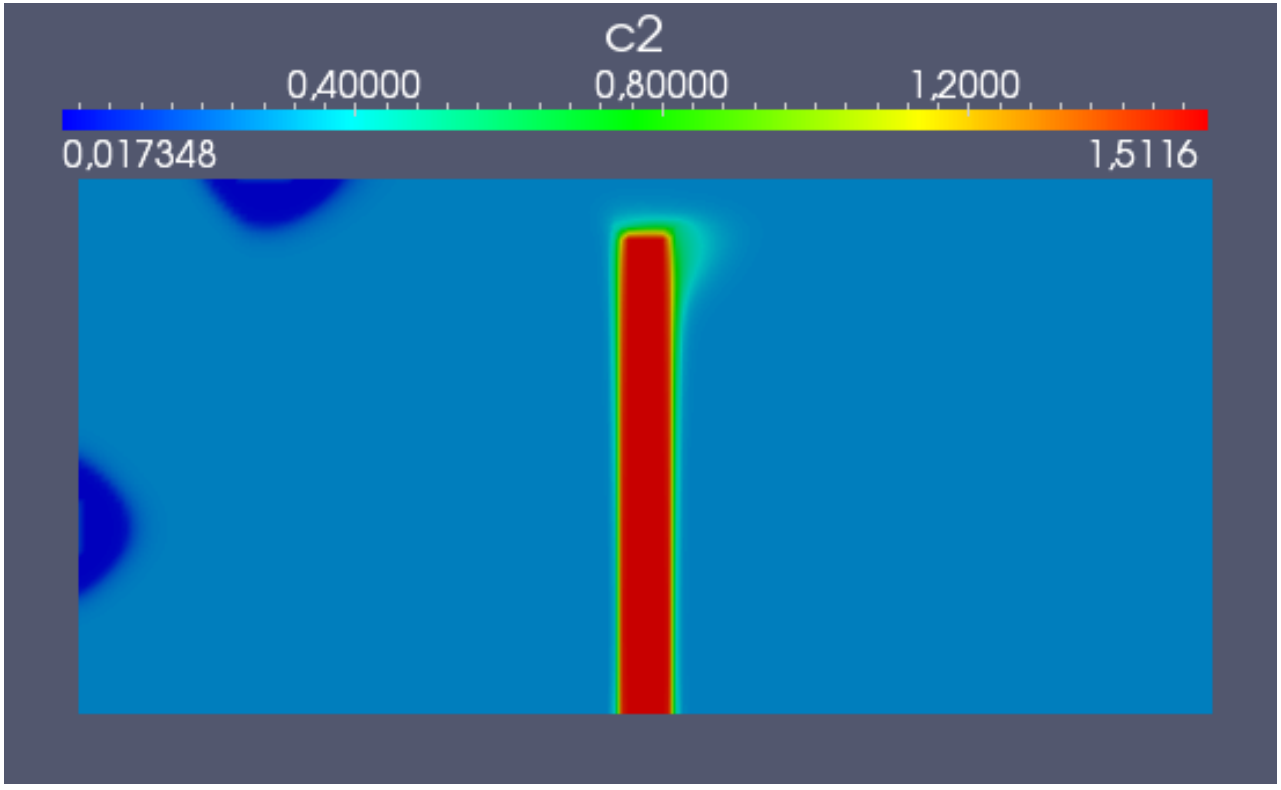
Initial conditions

	T_1	T_2	T_3	T_4	W
Medium A	0	-2	0	2	1
Medium B	0	-2	0	2	10

Inflow boundary conditions

	T_1	T_2	T_3	T_4	W
Injection $t \in [0, 5000]$	0.3	0.3	0.3	0	0
Leaching $t \in [5000, 6000]$	0	-2	0	2	0

Results of simulations



Computations done using GRT3D with a mesh of 80×168 cells

Numerical reactive transport model

Space discretization

with, for example, a finite difference method using N_m points

$$\begin{cases} T = (T_1, \ldots, T_k, \ldots, T_{N_m}), \\ T^T = (T_1^T, \ldots, T_j^T, \ldots, T_{N_c}^T) \end{cases}$$

Semi-discrete reactive transport model

$$\begin{cases} \omega \frac{dT_j^T}{dt} + LC_j^T(c) = Q_j^T, \quad j = 1, \ldots, N_c, \\ Tc_j(c,s) - T_k = 0, \quad k = 1, \ldots, N_m, \\ Ts_j(c,s) - W_k = 0, \quad k = 1, \ldots, N_m, \\ \text{initial condition for } T, \text{positivity constraints } c \geq 0, s \geq 0 \end{cases}$$

Numerical reactive transport model

Space discretization

with, for example, a finite difference method using N_m points

$$\begin{cases} T = (T_1, \ldots, T_k, \ldots, T_{N_m}), \\ T^T = (T_1^T, \ldots, T_j^T, \ldots, T_{N_c}^T) \end{cases}$$

Semi-discrete reactive transport model

$$\begin{cases} \omega \frac{dT_j^T}{dt} + LC_j^T(c) = Q_j^T, \quad j = 1, \ldots, N_c, \\ Tc_j(c,s) - T_k = 0, \quad k = 1, \ldots, N_m, \\ Ts_j(c,s) - W_k = 0, \quad k = 1, \ldots, N_m, \\ \text{initial condition for } T, \text{positivity constraints } c \geq 0, s \geq 0 \end{cases}$$

DAE formulation

$$\begin{cases} \omega \frac{d\mathbf{vec}\, T}{dt} + (L \otimes I)\mathbf{vec}\, C(c) - \mathbf{vec}\, Q = 0, \\ \mathbf{vec} \begin{pmatrix} Tc(c,s) \\ Ts(c,s) \end{pmatrix} - \mathbf{vec} \begin{pmatrix} T \\ W \end{pmatrix} = 0 \end{cases}$$

DAE Global approach with substitution

Time discretization: BDF scheme

$$\frac{d\mathbf{vec}\, T}{dt} \simeq \frac{a}{\Delta t}\mathbf{vec}\, T + \frac{1}{\Delta t}\mathbf{vec}\, Z,$$

$$\begin{cases} \frac{a\omega}{\Delta t}\mathbf{vec}\, T + (L \otimes I)\mathbf{vec}\, C(c) - \ldots = 0, \\ -(I \otimes N)\mathbf{vec}\, T + \mathbf{vec}\, \Phi(c, s) - \ldots = 0 \end{cases}$$

DAE Global approach with substitution

Time discretization: BDF scheme

$$\frac{d\mathbf{vec}\, T}{dt} \simeq \frac{a}{\Delta t}\mathbf{vec}\, T + \frac{1}{\Delta t}\mathbf{vec}\, Z,$$

$$\begin{cases} \frac{a\omega}{\Delta t}\mathbf{vec}\, T + (L \otimes I)\mathbf{vec}\, C(c) - \ldots = 0, \\ -(I \otimes N)\mathbf{vec}\, T + \mathbf{vec}\, \Phi(c,s) - \ldots = 0 \end{cases}$$

Substitution

$$R(c,s) = \frac{\Delta t}{a\omega}(L \otimes N)\,\mathbf{vec}\, C(c) + \mathbf{vec}\, \Phi(c,s) - \ldots$$

The Jacobian of R is

$$J_R(c) = \frac{\Delta t}{a\omega}(L \otimes N)\,\mathbf{diag}\,(\frac{dC}{dc}(c_j)) + \mathbf{diag}(J_\Phi(c_j, s_j)).$$

DAE Global approach with substitution

Time discretization: BDF scheme

$$\frac{d\mathbf{vec}\, T}{dt} \simeq \frac{a}{\Delta t}\mathbf{vec}\, T + \frac{1}{\Delta t}\mathbf{vec}\, Z,$$

$$\begin{cases} \frac{a\omega}{\Delta t}\mathbf{vec}\, T + (L \otimes I)\mathbf{vec}\, C(c) - \ldots = 0, \\ -(I \otimes N)\mathbf{vec}\, T + \mathbf{vec}\, \Phi(c,s) - \ldots = 0 \end{cases}$$

Substitution

$$R(c,s) = \frac{\Delta t}{a\omega}(L \otimes N)\,\mathbf{vec}\, C(c) + \mathbf{vec}\, \Phi(c,s) - \ldots$$

The Jacobian of R is

$$J_R(c) = \frac{\Delta t}{a\omega}(L \otimes N)\,\mathbf{diag}\,(\frac{dC}{dc}(c_j)) + \mathbf{diag}(J_\Phi(c_j, s_j)).$$

Nonlinear system

$$R(c,s) = 0$$

solved with Newton method

Choice of variables

Difficulties with variables (c, s)

- The derivatives $\frac{d\alpha}{dc}$, etc, are not easy to compute
- The positivity constraints must be satisfied at each time step

Choice of variables

Difficulties with variables (c, s)

- The derivatives $\frac{d\alpha}{dc}$, etc, are not easy to compute
- The positivity constraints must be satisfied at each time step

Change of variables

- assuming $c > 0, s > 0$, use $(\hat{c}, \hat{s}) = (\log(c), \log(s))$
- The positivity constraints are satisfied
- The Jacobian is easy to compute

Choice of variables

Difficulties with variables (c, s)

- The derivatives $\frac{d\alpha}{dc}$, etc, are not easy to compute
- The positivity constraints must be satisfied at each time step

Change of variables

- assuming $c > 0, s > 0$, use $(\hat{c}, \hat{s}) = (\log(c), \log(s))$
- The positivity constraints are satisfied
- The Jacobian is easy to compute

Difficulties with variables $(\log(c), \log(s))$

- Concentrations can become very small
- The matrix becomes almost singular for a component with almost null concentration

GRT3D software

Transport modules

- The velocity v is computed with MODFLOW
- The transport operator L is computed with MT3D

GRT3D software

Transport modules

- The velocity v is computed with MODFLOW
- The transport operator L is computed with MT3D

Chemistry modules

- The functions $\Phi(c, s)$ and $C(c)$
- The derivatives $J_\Phi(c, s)$ and $dC(c)/dc$

GRT3D software

Transport modules

- The velocity v is computed with MODFLOW
- The transport operator L is computed with MT3D

Chemistry modules

- The functions $\Phi(c, s)$ and $C(c)$
- The derivatives $J_\Phi(c, s)$ and $dC(c)/dc$

Coupling modules

- The function $R(c, s)$
- The derivative $J_R(c, s)$

GRT3D software

Transport modules

- The velocity v is computed with MODFLOW
- The transport operator L is computed with MT3D

Chemistry modules

- The functions $\Phi(c, s)$ and $C(c)$
- The derivatives $J_\Phi(c, s)$ and $dC(c)/dc$

Coupling modules

- The function $R(c, s)$
- The derivative $J_R(c, s)$

Solving modules

- The DAE solver IDA in SUNDIALS using Newton-LU method
- The sparse linear solver UMFPACK

Versions of GRT3D

GRT3D: First version with logarithms

Logarithmic variables $\log(c), \log(s)$
No elimination of T and C

Versions of GRT3D

GRT3D: First version with logarithms

Logarithmic variables $\log(c), \log(s)$
No elimination of T and C

GRT3DRL: Reduced version with logarithms

Logarithmic variables $\log(c), \log(s)$
Elimination of T and C in the linearized equations

Versions of GRT3D

GRT3D: First version with logarithms

Logarithmic variables $\log(c), \log(s)$
No elimination of T and C

GRT3DRL: Reduced version with logarithms

Logarithmic variables $\log(c), \log(s)$
Elimination of T and C in the linearized equations

GRT3DRSL: Reduced version without logarithms

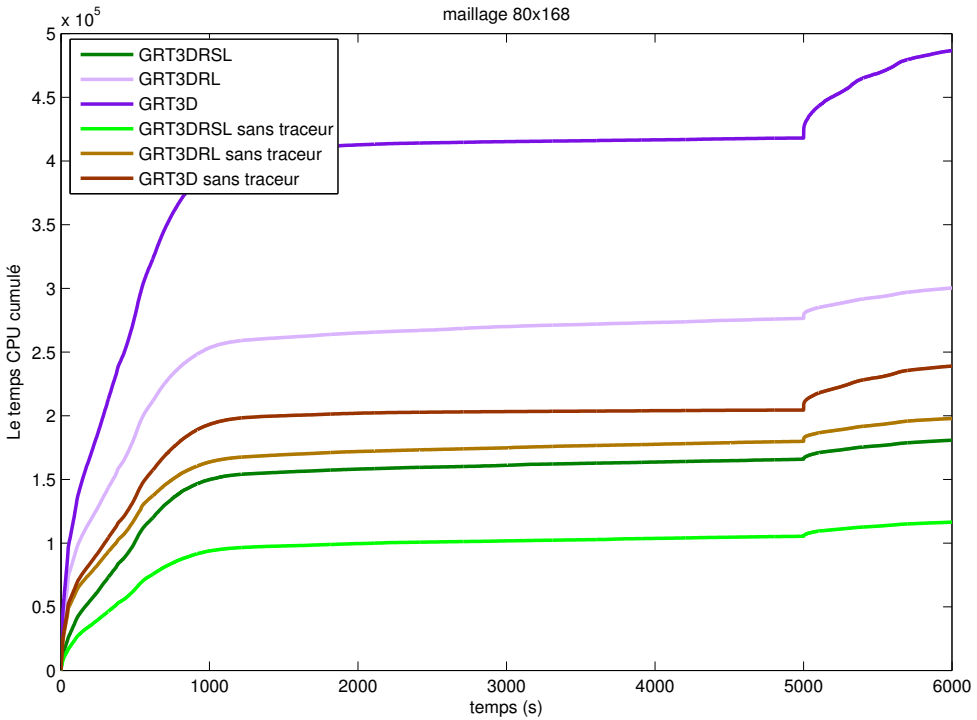
Variables c, s
Elimination of T and C in the linearized equations

Memory requirements

System size

Mesh	GRT3D with c_1	GRT3DRL with c_1	GRT3D without c_1	GRT3DRL without c_1
N_m	$13N_m$	$5N_m$	$10N_m$	$4N_m$
20x42	10920	4200	8400	3360
40x84	43680	16800	33600	13440
80x168	174720	67200	134400	53760

CPU time

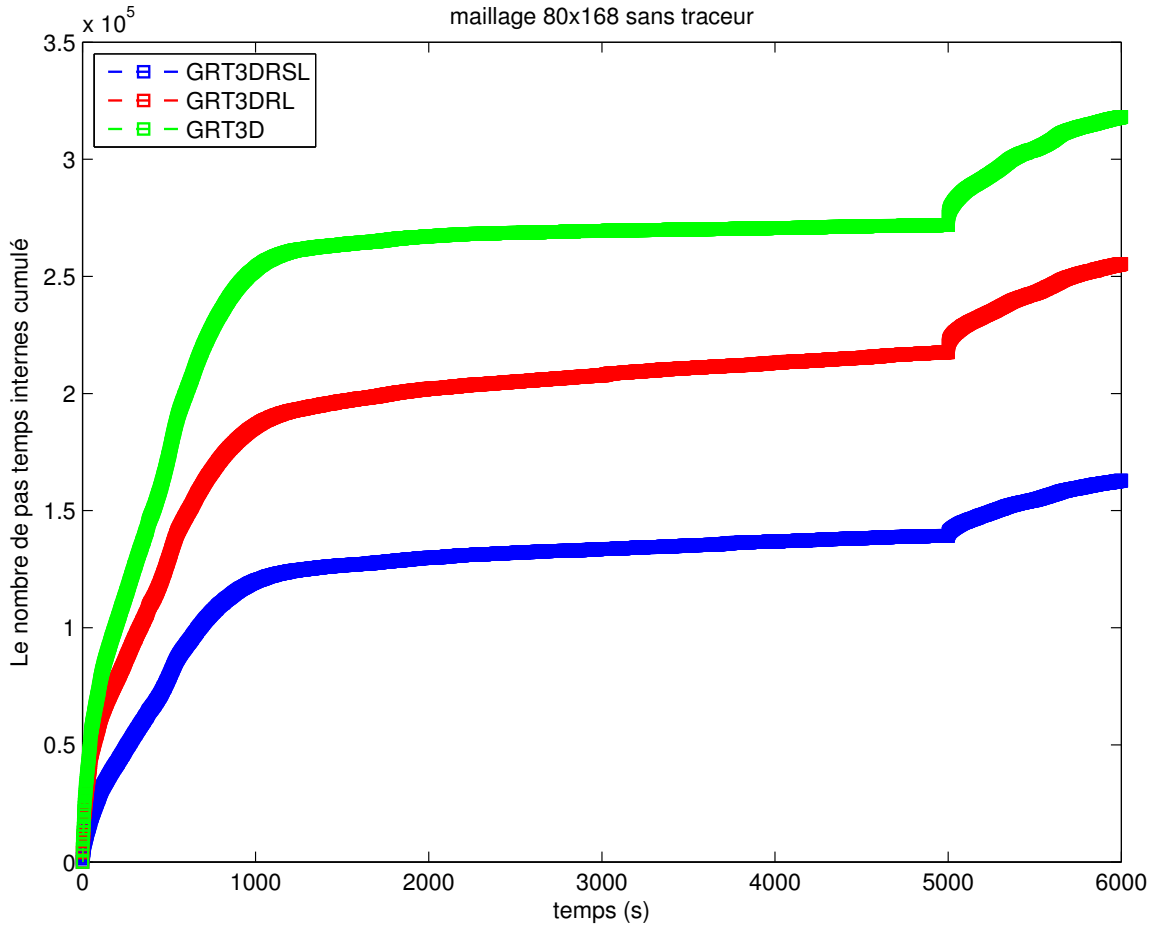


Removing the inert component c_1 reduces the CPU time

Reducing the system size is efficient

Using variables c, s is faster than using logarithmic variables $\log(c), \log(s)$

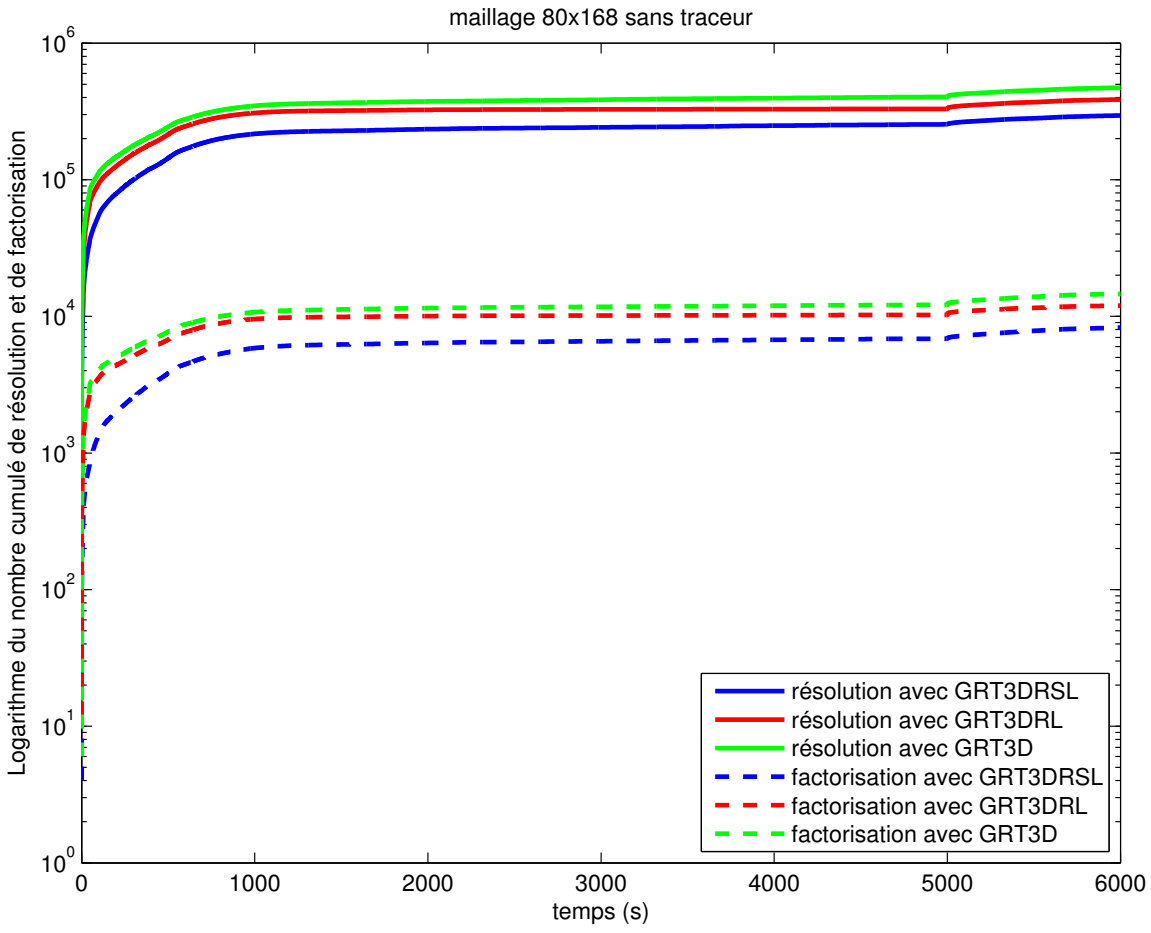
Number of time steps



Many time steps until times $t = 1000$ and near $t = 5000$

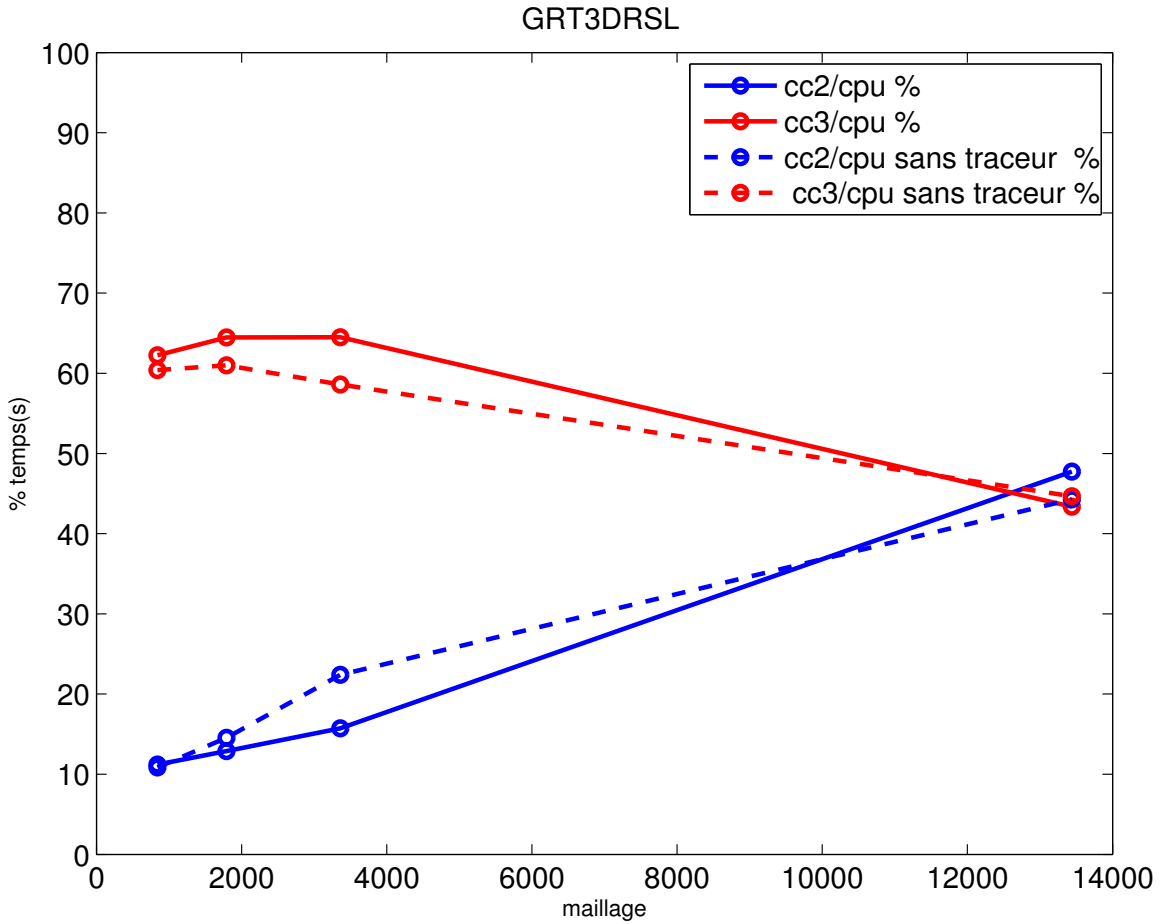
The CPU time is directly correlated to the number of time steps

Number of LU factorizations



Modified Newton method reduces efficiently the number of factorizations

CPU time of factorization and solving



Linearized equations use 90 % of total CPU time (fine mesh 80×168)

Conclusion

Summary: accuracy and efficiency

- DAE global approach (implicit scheme and Newton method)
- Efficiency of system size reduction
- Efficiency of adaptive time step and modified Newton iterations
- Logarithmic variables may lead to ill-conditioned systems

Conclusion

Summary: accuracy and efficiency

- DAE global approach (implicit scheme and Newton method)
- Efficiency of system size reduction
- Efficiency of adaptive time step and modified Newton iterations
- Logarithmic variables may lead to ill-conditioned systems

Future work

- Iterative parallel linear solver
- Parallel chemistry computations
- Precipitation-dissolution with vanishing p
- Adaptive mesh refinement