NUMERICAL METHODS FOR SOLVING NONLINEAR SYSTEMS ARISING IN REACTIVE TRANSPORT MODELLING

BOUILLARD N.[1], HERBIN R.[2], MONTARNAL P. [1]

[1]CEA Saclay, DANS/DM2S/SFME/MTMS, 91191 Gif sur Yvette Cedex, France.
[2]Université de Provence, 39 Rue Joliot Curie, 13453 Marseille 13.

Abstract

In this paper, a nonlinear conjugate gradient algorithm is studied and used for the numerical solution of reactive transport problems. This method is derived from a general framework [Fokkema et al., 1998] for the resolution of nonlinear systems. This derivation suggests several directions to improve the method. Numerical applications show and discuss the advantages of this method compared with the Picard and Newton methods.

1. INTRODUCTION

Classical approaches for solving nonlinear systems provided by reactive transport problems are typically of the Picard and Newton type. The former includes sequential iterative approaches and sequential non-iterative approaches [Yeh and Tripathi, 1989]. The latter is often used in global implicit approaches when reactive transport equations system is considered as a whole. Equilibrium equations are then often treated by direct substitution approaches (DSA) [Fahs et al., 2005]. The interest is the direct access of derivatives and the drawback is the size of the linear system. Some authors work in reducing the size of the system by cleverly combining the equations in a different way than DSA [Krautle and Knabner, 2005]. For both approaches, when considering dissolution-precipitation reactions, it should be mentioned that the derivatives are not as easy to obtain.

Our work is included in a software platform project Alliances [Montarnal et al., 2006]. The goal is to improve iterative sequential algorithm keeping a code coupling approach [Bouillard, 2006].

In this paper, two discrete formulations of the reactive transport as a nonlinear system are proposed (Section 2). Then a general algorithm is described (Section 3) allowing the derivation of a nonlinear conjugate gradient algorithm (Section 4). Finally, numerical applications allow to compare this algorithm with more classical approaches using the Picard and Newton method (Section 5).

2. FORMULATION OF REACTIVE TRANSPORT PROBLEMS

2.1. **Description and discretization.** An accurate description of reactive transport model may be found in several works [Yeh and Tripathi, 1989] [Krautle and Knabner, 2005] [Holstad, 2000] and references therein. A reactive transport model should describe the spatial and temporal evolution of a set of chemical species under the influence of

transport phenomena and chemical reactions. The system of equations consists of PDEs for the mobile species and ODEs for the immobile ones, all of them coupled through reaction terms. The characteristic time scales of the different reactions may cover a large range. Thus, it is desirable to model some of them with kinetics and other with equilibrium, the latter leading to algebraic equations.

The following assumptions are made in this paper, (1) the porosity ω is assumed to be constant, (2) the flow is stationary and the porous media is saturated, (3) the dispersion-diffusion tensor D does not depend on the species, (4) transport phenomena only affect species in liquid phase and (5) all chemical processes are assumed to be under equilibrium. The latter is often refered to as local equilibrium assumption.

Independent algebraic equations are linked relations that are used to reduce the number of unknowns thus defining components. Hence, the set of differential algebraic equations may be written as [Yeh and Tripathi, 1989]

$$\left\{\begin{array}{lcl} \omega\partial_t T_e - L_e(C_e) & = & 0 \\ C_e & = & \Psi_e(T_1,\cdots,T_e,\cdots,T_{nc}) \\ T_e & = & C_e + F_e \end{array}\right. \tag{1}$$

where $e = 1,\cdots,nc$, with nc the number of component species, $T = (T_1,\cdots,T_{nc})^T$ the total analytical concentration, C the total dissolved concentration, F the total immobile concentration, $L_e C_e = \nabla\cdot(D\nabla C_e - C_e\vec{u})$ (where $\vec{u}$ is the Darcy velocity) the linear transport operator and $\Psi = (\Psi_1,\cdots,\Psi_{nc})^T$ is the nonlinear chemical operator which is computed by an external code in our simulations.

After discretization by the classical finite volume method [Gallouët et al.] on admissible meshes with an Euler implicit time scheme and an upwind scheme for the convective term, the system (1) reads:

$$\left\{\begin{array}{lcl} \omega\frac{SF}{\Delta t}\mathbf{T}^{n+1} - \mathbf{L}(\mathbf{C}^{n+1}) & = & s^n \\ \mathbf{C}^{n+1} & = & \Psi(\mathbf{T}^{n+1}) \\ \mathbf{T}^{n+1} & = & \mathbf{C}^{n+1} + \mathbf{F}^{n+1} \end{array}\right. \tag{2}$$

where s^n depends on the boundary conditions and the previous time step and $\frac{SF}{\Delta t}$ is the ratio of mesh element areas SF by time step Δt.

2.2. **Two formulations.** Two formulations are proposed. At each time step, by rearranging Eqs (2), the problem is either to find a zero of $R_{\mathbf{C}}$ in the $\mathbf{C}$ formulation (Eqn (3)) or a zero of $R_{\mathbf{T}}$ in $\mathbf{T}$ formulation (Eqn (4)):

$$R_{\mathbf{C}}(c) = c - \Psi\left((\omega\frac{SF}{\Delta t})^{-1}(s^n + \mathbf{L}c)\right) \tag{3}$$

$$R_{\mathbf{T}}(\theta) = \omega\frac{SF}{\Delta t}\theta - \mathbf{L}\Psi(\theta) - s^n. \tag{4}$$

Classical splitting algorithms solve either one of these two nonlinear systems (3), (4) or another nonlinear sytem under the form

$$MR_{\mathbf{T}}(\theta) = 0 \quad \text{or } MR_{\mathbf{C}}(c) = 0,$$

for some matrix M.

3. A GENERAL FRAMEWORK FOR SOLVING NONLINEAR PROBLEMS

At each time step, it is assumed that the problem is:

$$\text{find } x \in \mathbb{R}^N \text{ such that } R(x) = 0$$

where $R : \mathbb{R}^N \mapsto \mathbb{R}^N$ and N is the size of the system. In reactive transport, N is the number nm of mesh elements multiplied by the number of components nc.

3.1. **A general algorithm.** As previously mentionned, nonlinear systems of the reactive transport type, are often solved in two steps, (1) a linearization technique of the Newton type, and (2) an iterative method for solving the linear system. In practical situations, the dimension of the linear system is large, so that the iterative method is often a projection method (Krylov methods). One can try to invert the order of operations, (1) first a projection method which consists in building or choosing (at low cost) a family of P vectors and then (2) solving a nonlinear problem of size P, $1 \leq P << N$ either by the Newton method or another nonlinear method. The idea is to use the Newton method on

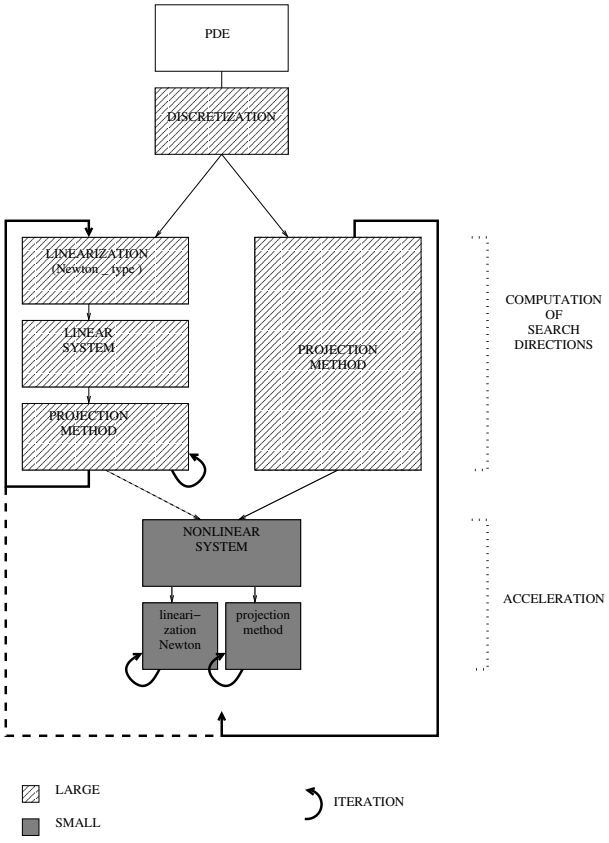

FIGURE 1. Nonlinear solvers. Inspired by [Weiss and Podgajezki, 1999] Left: the focus is on the computation of search directions, outer iterations of Newton type and inner iterations for a projection method in the iterative solver; It could be improved with acceleration techniques ; Right: focus on the acceleration technique, outer iterations for a projection method and inner iterations for solving a small nonlinear system either by a Newton or a projection method.

a smaller problem, in order to reduce the number of derivatives to be computed. This is particularly interesting in a code coupling approach where the computation of derivatives, by finite differences, requires several external code calls which affect the CPU time.

The nonlinear conjugate gradient methods stated in the next section is inspired by the second approach. Both approaches are outlined in Figure 1 inspired by [Weiss and Podgajezki, 1999]. For both situations, a general scheme is proposed in Algorithm 1 [Fokkema et al.,

1 Choose x_0 and compute $r_0 = -R(x_0)$.
2 $V_{-1} = [\ ]$, $W_{-1} = [\ ]$.
for $k = 0, 1, \cdots$ convergence **do**
3 *Compute an approximation J_k for the Jacobian matrix $R'(x_k)$*
4 *Solve the correction equation (approximately).* Compute an (approximate) solution p_k for the *correction equation* $J_k p = r_k$.
5 *Expand the search space.* Select a v_k in the span (V_{k-1}, p_k) that is linearly independent of V_{k-1} and update $V_k = [V_{k-1}|v_k]$.
6 *Expand the shadow space.* Select a w_k that is linearly independent of W_{k-1} and update $W_k = [W_{k-1}|w_k]$.
7 *Solve the projected problem (5) or the residual minimum problem (6).* In the latter case, the computation of a shadow space is not necessary. The selected solution is denoted by y_k.
8 *Update.* Update the approximation $x_k = x_k + V_k y_k$ and *Compute the residual* $r_k = -R(x_k)$.
9 *Restart.* If $dim(span(V_k)) > P$ then take a suitable combination of the columns of W_k and V_k of dimension P.
end

Algorithm 1: General nonlinear solver (AIN algorithm [Fokkema et al., 1998]).

1998]. Let

$$G_k(y) \quad = \quad R(x_k + V_k y).$$

The acceleration method can involve either a Galerkin condition, stated in Eqn (5):

$$W_k^H G_k(y) \quad = \quad 0, \tag{5}$$

where $V_k = [v_1, \cdots, v_k]$ is some basis of search space and W_k is some matrix of the same dimensions as V_k; or it can be replaced by a minimal residual (RM) condition (6):

$$\min_{y \in \mathbb{R}^k} \|G_k(y)\|_2, \tag{6}$$

or a mix of both called restricted minimal residual (RMR) condition:

$$\min_{y \in \mathbb{R}^k} \|W_k^H G_k(y)\|_2. \tag{7}$$

The Algorithm 1 is very general and includes well known techniques for solving linear and nonlinear system. It was introduced by [Fokkema et al., 1998] for accelerated inexact newton (AIN) methods. It provides a natural link between linear and nonlinear methods.

We just explain how to derive the classical Newton and Picard method in this framework.

3.2. **Newton method.** The step **3** is the computation of the Jacobian matrix. In code coupling approach, it is usually done by numerical derivation. Then the new step **4** is nothing but the Newton equation that can be solved iteratively. At each outer iteration, an accurate resolution of Newton equation is not necessary. Thus, some criteria might be used to poorly solve the Newton equation [Dembo et al., 1982] at first outer iterations and then be more accurate. In a simple Newton method, steps **5**, **6**, **7** and **9** are omitted. Remark that Jacobian-free Newton-Krylov methods also enter this framework.

Choose x_0 and compute $r_0 = -R(x_0)$.
for $k = 0, 1, \cdots$ convergence **do**
3 *Compute an approximation of the Jacobian matrix J_k by finite differences*
4 *Solve the correction equation (approximately) $J_k p_k = r_k$.*
8 *Update $x_k = x_k + p_k$ and compute the residual $r_k = -R(x_k)$.*
end

Algorithm 2: Newton algorithm

3.3. **Picard method.** In step **3**, J_k is approximated by a constant matrix, say the linear part of $R'(x_k)$. In our numerical tests, this matrix will be a block diagonal one:

$$I - (\omega \frac{SF}{\Delta t})^{-1}\mathbf{L} \text{ in } \mathbf{C} \text{ variable}, \quad \omega \frac{SF}{\Delta t} I - \mathbf{L} \text{ in } \mathbf{T} \text{ variable}. \tag{8}$$

Then, the step **4** appears to be a preconditioning one. Steps **5**, **6**, **7** and **9** are skipped and there remains only the update of the iterate and a residual computation in step **8**.

Choose x_0 and compute $r_0 = -R(x_0)$.
3 *Choose J_k by (8) (independent of k).*
for $k = 0, 1, \cdots$ convergence **do**
4 *Solve the correction equation (approximately) $J_k p_k = r_k$.*
8 *Update $x_k = x_k + p_k$ and compute the residual $r_k = -R(x_k)$.*
end

Algorithm 3: Picard algorithm

4. NONLINEAR CONJUGATE GRADIENT ALGORITHMS

4.1. **Derivation ($V_k = W_k, P = 1$).** Nonlinear conjugate gradient methods are broadly used in unconstrained optimization. Recently, global convergence of an algorithm without line search has been proved under mild assumptions [Sun and Zhang, 2001]. This result allows the use of this algorithm for solving nonlinear systems even if the convergence result does not hold. This type of algorithm was recently succesfully used in coupling geomechanical and reservoir codes [Daim et al., 2002].

The nonlinear conjugate gradient method is stated in Algorithm 4. It can be derived from the general Algorithm 1 in the following way. In step **3**, J_k is approximated by a

constant matrix as in the Picard approach (8). If R were linear, J_k would be a preconditioner of the linear system. Then, in step **5** to **8**, one chooses $V_k = W_k$ and dim(V_k)=1. In step **5**, the search direction v_k is computed by

$$\begin{cases} v_k = p_k & \text{if } k = 0 \\ v_k = p_k + \beta_k v_{k-1} & \text{if } k > 0, \end{cases} \tag{9}$$

where β_k is the positive part of the Polak-Ribière formula

$$\beta_k = \max(\frac{(r_k - r_{k-1})^H p_k}{r_{k-1}^H p_{k-1}}, 0)$$

which is the most efficient in practice. The positive part allows automatic restart. Eqn (9) is the classical conjugate gradient formula for updating search directions. Given the assumptions, the *projected problem* of step **7** writes

$$0 = v_k^H G_k(y) = v_k^H R(x_k + y v_k)\,, y \in \mathbb{R}. \tag{10}$$

If R is linearized by

$$R(x_k + y v_k) \simeq R(x_k) + y R'(x_k) v_k$$

then the exact solution of (10) is

$$y_k^{proj} := \frac{-R(x_k)^H v_k}{v_k^H (R'(x_k) v_k)}$$

where $R'(x_k)v_k$ is computed with a numerical derivation

$$R'(x_k) v_k = \frac{R(x_k + \varepsilon v_k) - R(x_k)}{\varepsilon}$$

in a code coupling approach.

Finally, if an MR (resp. RMR) approach is chosen, Eqn (10) is replaced by (6) (resp. (7)) which solution is easily computed

$$y_k^{MR} := \frac{-R(x_k)^H (R'(x_k) v_k)}{\|R'(x_k) v_k\|_2^2} \quad (\text{ resp. } y_k^{RMR} := \frac{-v_k^H R(x_k)}{v_k^H R'(x_k) v_k} = y_k^{proj} \text{ in this case}).$$

Choose x_0 and compute $r_0 = -R(x_0)$
3 *Choose J_k by (8) (independent of k).*
for $k = 0, 1, \cdots$ convergence **do**
4 *Solve the correction equation $J_k p_k = r_k$*
5 **if** $k = 0$, $v_0 = p_0$ **else** $\rho_+ = (r_k - r_{k-1})^H p_k$, $\beta_k = max(\rho_+/\rho_-, 0)$ *and* $v_k = p_k + \beta_k v_{k-1}$ **end**.
7 $q_+ = r_k^H v_k$, $u = R'(x_k)v_k$ or an approximation by numerical derivation, $q_- = v_k^H u$, $y_k = q_+/q_-$.
8 $x_k = x_k + y_k v_k$, $\rho_- = r_k^H p_k$, $r_{k-1} = r_k$, $r_k = -R(x_k)$.
end

Algorithm 4: Nonlinear conjugate gradient algorithm with projection method

4.2. **Improvements ($V_k \neq W_k$ and/or $P \geq 1$).** At least, The nonlinear algorithm should degenerate in an efficient linear method when chemistry is linear. For nonlinear systems provided by reactive transport, the transport matrix is nonsymmetric if convection is taken into account. In this case, linear conjugate gradient method is not efficient. In BiCG type approaches [van der Vorst, 2003] for solving

$$R(x) = 0 \text{ with } R(x) = Ax - b \text{ and } A \text{ nonsymmetric}$$

the idea is to use two different matrix W_k and V_k and solve the projection problem:

$$W_k^H R(x_k + V_k y) = W_k^H \left(-r_k + AV_k y\right) = 0\,, y \in \mathbb{R}^k$$

where W_k spans the Krylov space $K^{k-1}(A^H, r_0)$ and V_k spans $K^{k-1}(A, r_0)$. Besides,

$$v_k^H w_j = 0 \text{ if } j < k, \quad v_j^H w_k = 0 \text{ if } j < k \text{ (bi orthogonality conditions).}$$

So in nonlinear applications, it seems natural to also choose $W_k \neq V_k$ for generalizing BiCG type approaches. An initial black-box approach consists in formally replacing matrix vectors products by numerical derivation in the BiCGStab algorithm for linear systems. Some authors have begun to work around an extension of BiCGStab [Graves-Morris].

Another promising possibility is to choose $P \geq 2$. Indeed, it has been observed [Weiss and Podgajezki, 1999] on some numerical applications that the switch $P = 1$ to $P = 2$ results in considerable acceleration.

5. NUMERICAL APPLICATIONS

In what follows, we consider the **C** formulation which was observed on numerical tests to be more efficient than the **T** formulation.

5.1. **A priori discussion.** In Table 1, the main steps of each algorithm are reported.

Steps	res	$LS(size)$	$x + y$	$x^H y$
Picard (PIC), Algorithm 3 p. 5				
4		$nc\ LS(nm)$		
8	1		1	
Nonlinear preconditioned conjugate gradient (PCG), Algorithm 4 p. 6				
4		$nc\ LS(nm)$		
5	1		1	1
7			1	2
8	1			1
Newton (NEW), Algorithm 2 p. 5				
3	nc			
4		$LS(nm \cdot nc)$		
8	1		1	1

TABLE 1. Main steps for one outer iteration. res is the number of residual computation, it corresponds to one call to the chemistry code, $LS(size)$ denotes an inversion of a linear system, for PIC and PCG the cost is this of choice (8).

The unit cost of an algorithm $C_u(ALG)$ is defined as the CPU time of one outer iteration of algorithm ALG. The average number of outer iterations needed to reach convergence

during one time step is N_i. Global cost $C_g(ALG)$ is defined by:

$$C_g(ALG) = C_u(ALG) \times N_i(ALG) \times (\text{number of time steps}),$$

which is the CPU time of a simulation run. From Table 1, if inner products are neglected, $C_u(PCG) = C_u(PIC) + 1$ res.

5.2. **Applications(diffusion only,$nc = 5$ ou 9, 2D space, square mesh).** Table 2 reports the comparisons of three algorithms on a leaching case of Portlandite which occurs under the action of alkalin water [Read and Falck, 1996]. It is a 2D extension in of a 1D case. In Table 3, comparisons are shown for a second test case involving

ALG	C_g	N_i	$C_u/C_u(PIC)$	tot it	C_g	N_i	$C_u/C_u(PIC)$	tot it	C_g	N_i	$C_u/C_u(PIC)$	tot it
PCG	5.8	7.9	1.85	993	3.4	17.9	1.83	627	3.0	21.3	1.82	555
PIC	7.0	17.8	1	2219	7.8	75.3	1	2634	8.2	106.4	1	2766
NEW	6.0	1.3	12	159	2.5	2.0	11.9	71	1.9	2.0	12.1	53
$\frac{\Delta t}{\Delta t_0}$			$(0.1:1)$				$(0.1:5)$				$(0.1:10)$	
nb Δt			125				35				26	

TABLE 2. A Portlandite leaching case. $nc = 5$, $nm = 160$, only diffusion, 2D space. The sequence of time steps is smoothly increasing inside the specified range with $\Delta t_0 = \frac{D}{\Delta x^2}$, C_g in minutes.

the dissolution of uraninite and redox reactions under the infiltration of rain water in an aquifer [De Windt et al., 2003]. In both cases, it is observed that $C_u(PCG) \simeq 2 \cdot C_u(PIC)$,

ALG	C_g	N_i	$C_u/C_u(PIC)$	tot it	C_g	N_i	$C_u/C_u(PIC)$	tot it
PCG	24.7	7.6	2	1566	33.5	12.9	2.1	2059
PIC	28.1	17.2	1	3522	32.4	26.1	1	4178
NEW	45.7	1.3	21.4	268	38.25	1.4	22	225
$\frac{\Delta t}{\Delta t_0}$			$(0.1:15)$				$(0.1:20)$	
nb Δt			205				160	

TABLE 3. A Uraninite leaching case. $nc = 9$, $nm = 300$, only diffusion, 2D space. The sequence of time steps is smoothly increasing inside the specified range $\Delta t_0 = \frac{D}{\Delta x^2}$, C_g in minutes.

thus it does not depend on nc contrary to $C_u(NEW)$. Indeed, $C_u(NEW) \simeq 12 \cdot C_u(PIC)$ in Table 2 ($nc = 5$) and $C_u(NEW) \simeq 22 \cdot C_u(PIC)$ in Table 3 ($nc = 9$) ; this is due to the increasing cost of steps **3** and **4** when nc is growing. In both cases, it is observed that $N_i(NEW) < N_i(PCG) < N_i(PIC)$.

The main drawback of PIC is observed in the two Tables: you do not save CPU time when you increase Δt since $C_g(PIC)$ do not decrease. On the contrary, in Table 2, $C_g(PCG)$ and $C_g(NEW)$ decrease when Δt increases. So they perform better than PIC. As for the global cost, the NEW method performs better than the two other ones in the first case and has poor results in the second one. Indeed $C_u(NEW)/C_u(PIC)$ is too large. The global cost of PCG is betweeen PIC and NEW in both cases. Performance of PCG in Table 3 should be improved since parameters like convergence threshold and the maximum number of iterations per time step might be adjusted with more care.

Indeed, we only have $N_i(PCG) = 1/2 \cdot N_i(PIC)$ whereas in Table 2, it was $N_i(PCG) \simeq 1/5 \cdot N_i(PIC)$. Note that C_g is quite sensitive to the implementation whereas E_i is not.

5.3. **Conclusion.** First results show that nonlinear conjugate gradient methods are promisive for solving nonlinear systems. Still, this method is not designed for treating problems where the transport operator $\mathbf{L}$ is not symmetric. So, we plan to extend and improve the method as proposed in paragraph 4.2. Another direction is to improve the choice of the inital iterate x_0 and propose efficient time stepping strategies. At last, it should be desirable to find out efficient preconditioning techniques which is more challenging.

References

[Bouillard] N. Bouillard (2006), Ph.D. Thesis in preparation.

[Daim et al.] F.Daim, R.Eymard, D.Hilhorst, M.Mainguy and R.Masson (2002), A Preconditioned Conjugate Gradient Based Algorithm for Coupling Geomechanical-Reservoir Simulations, *Oil and Gas Science and Technology-Rev*, *57*, IFP.

[Dembo et al.] R.S. Dembo, S.C. Eisenstat, T. Steihaug (1982), Inexact Newton methods, *SIAM J. Numer. Anal.*, *19*, pp. 400–408.

[De Windt et al.] L. De Windt, A. Burnol, P. Montarnal, J. van der Lee (2003), Intercomparison of reactive transport models applied to UO_2 oxydative dissolution and uranium migration, *J. of Contaminant Hydrology*, *61*, pp. 303–312.

[Fahs et al.] M. Fahs, P. Ackerer, J. Carrayrou (2005), Comparison of Two Formulations for Multicomponent Reactive Transport in Porous Media Modeling by global approach, *SIAM Geosciences, Avignon, France.*

[Fokkema et al.] D.R. Fokkema, G.L.G. Sleijpen, H. van der Vorst (1998), Accelerated inexact newton schemes for large systems of nonlinear equations, *SIAM J. Sci. Comput.*, *19(2)*, pp. 657–674.

[Gallouët et al.] T. Gallouët, R. Eymard, R. Herbin (2000), Finite Volume Method, Handbook for Numerical Analysis, *P.G. Ciarlet, J.L. Lions Eds, North Holland.*

[Graves-Morris] P.R. Graves-Morris , BiCGStab, VPAStab and an adaptation to mildly nonlinear systems, *submitted to J. of Computational and Applied mathematics.*

[Holstad] A. Holstad (2000), A mathematical and numerical model for reactive fluid flow systems, *Computational Geosciences*, *4*, pp. 103–139.

[Krautle and Knabner] S. Krautle, P. Knabner (2005), A new numerical reduction scheme for fully coupled multicomponent transport-reaction problems in porous media, *Water Resources Research*, *41*, W09414.

[Montarnal et al.] P. Montarnal, A. Bengaouer, L. Loth, C. Chavant (2006), Alliances : simulation platform for radioactive waste disposal, *CMWR XVI - Computational Methods in Water Resources Copenhaguen, Denmark.*

[Read and Falck] D. Read, W. Falck (1996), Chemval2: a coordinated research initiative for evaluating and enhancing chemical models in radiological risk assessment, *Vol. EUR 16648-EN of Nuclear Science and Technology EC Series.*

[Sun and Zhang] J. Sun, J. Zhang (2001), Global convergence of conjugate gradient methods without line search, *Annals of Operations Research*, *103*, pp. 161–173.

[van der Vorst] H. van der Vorst (2003), Iterative krylov methods for large linear systems, *Cambridge University Press.*

[Weiss and Podgajezki] R. Weiss, I. Podgajezki (1999), Overview on new solvers for nonlinear systems, *Appl. Numer. Math.*, *30(2-3)*, pp. 379–391.

[Yeh and Tripathi] G.T. Yeh, V.S. Tripathi (1989), A Critical Evaluation of Recent Developments in Hydrogeochemical Transport Models of Reactive Multichemical Components, *Water Resources Research, 25(1)*, pp. 93–108.