Algorithms for activity correction models for reactive transport.

Jerome Carrayrou[1], Caroline Bertagnolli[1], and Marwan Fahs[1]

[1]University of Strasbourg

October 13, 2020

Abstract

Reactive transport codes are very useful elements of environmental research. They now contain multiphysics with very complex algorithms, including flow, transport, chemical and sometimes heat transport, mechanical and/or biological algorithms. Because of this complexity, some parts of these algorithms still have not been sufficiently studied. Here, we present a comparison of 3 algorithms for activity correction, a specific subset of equilibrium chemistry algorithms. We show that the most used algorithm (the inner fixed-point algorithm) or the most rigorous algorithm (the full Newton) might not be the most efficient, and we propose the outer fixed-point algorithm, which is more robust and faster than other algorithms.

Algorithms for activity correction models for reactive transport.

Jérôme Carrayrou[1,*], Caroline Bertagnolli[2], Marwan Fahs[1]

[1] University of Strasbourg, Laboratoire d'Hydrogéologie et de Géochimie de Strasbourg; UMR 7517 UdS-CNRS

[2] *University of Strasbourg, Reconnaissance et Procédés de Séparation Moléculaire, IPHC; UMR 7178 UdS-CNRS*

[*] *corresponding author: jerome.carrayrou@unistra.fr*

Keywords

Geochemical modelling; instantaneous equilibrium chemistry; activity correction, outer fixed-point algorithm, Davis activity, Debye-Hückel activity, B-dot model.

Abstract

Reactive transport codes are very useful elements of environmental research. They now contain multiphysics with very complex algorithms, including flow, transport, chemical and sometimes heat transport, mechanical and/or biological algorithms. Because of this complexity, some parts of these algorithms still have not been sufficiently studied. Here, we present a comparison of 3 algorithms for activity correction, a specific subset of equilibrium chemistry algorithms. We show that the most used algorithm (the inner fixed-point algorithm) or the most rigorous algorithm (the full Newton) might not be the most efficient, and we propose the outer fixed-point algorithm, which is more robust and faster than other algorithms.

Introduction

The problem of groundwater management has received increasing attention, and many tools have been developed to address this issue. One of these tools, reactive transport modeling, was first limited to laboratory experiments [1] and then extended to the comprehension of problems in various fields [2]. Reactive transport modeling is actually a mature research field that has produced important results in many environmental

domains, such as water management, sea water intrusion [3], long-term nuclear waste storage [4] and heavy metal contamination [5]. Numerous reactive transport codes are available, and some review articles [6–12] propose an overview of them. Examining these articles, it can be seen that all these codes include one or more activity correction models. Even though the different models of activity correction are usually well-detailed, the algorithmic method used to compute the activity coefficients and to incorporate these calculations into the entire chemistry algorithm is not given.

We show here that different algorithms can handle activity corrections and that they are not all equivalent. It seems that the most used algorithm, named the inner fixed-point algorithm, leads to numerical instabilities when handling highly-charged chemical species. We then recommend the new algorithm presented in this work: the outer fixed-point algorithm. It is more robust, faster and less sensitive to the initial conditions.

General concepts

A general formulation of a chemical reaction leading to the formation of one of the Nc species (C_i) from the number Nx of chosen components X_j is written as:

Instantaneous equilibrium chemistry is usually described using two fundamental concepts: mass conservation equations and mass action laws. According to the classical formulation stated by Morel and Morgan, mass conservation equations describe the conservation of the total concentrations of the components (T_j), and mass action laws describe the formation of each chemical species as a combination of the Nx chosen components.

Mass conservation equations are written using the species concentrations [C_i]:

On the other hand, mass action laws are written using the species $\{C_i\}$ and components $\{X_j\}$ of the activities:

To ensure the closure of the system, an activity model is used. The activity coefficient ($_i$) is less than one and determines the species activity from its concentration.

Several activity models have been developed, all of which use the ionic strength of the solution (I):

Davis model

One of the most used activity models for reactive transport is the Davis model, where the activity coefficients are given by:

The parameters A and B are defined by:

et

Extended Debye-Hückel model

Another activity model commonly used in reactive transport is the extended Debye-Hückel model. In this case, the activity coefficients are computed by:

B-dot model

The purpose of the B-dot model is to give nonunit activity coefficients for neutral chemical species.

Newton algorithm

The methodology for computing chemical equilibrium is well established [13–19] and is always built on a Newton procedure. By incorporating the Nc mass action laws into the Nx conservation equation, one can obtain an Nx by Nx nonlinear system that must be iteratively solved. Here, we present the approach where mass action laws are written in a logarithmic form.

We define the logarithm of the components activity as:

This can be rewritten as a matrix formulation:

The mass action laws are then:

These can be rewritten by using the vector of species concentrations :

The conservation equations are:

Combining both equations, we obtain:

This nonlinear system is iteratively solved using a Newton procedure. At the nth iteration, the error is given by:

By computing the derivative of the error versus the logarithm of the components of the activity, we obtain the Jacobian matrix . By solving the linear system :

Then, a new value of the activity component is given by:

The procedure is repeated until the error is sufficiently small.

Possible algorithms

Historically, activity correction has been neglected when computing the Jacobian matrix. This approximation is justified if:

1. The ionic strength is small enough so that the activity coefficients can be assumed to be equal to one (ideal solution).
2. The changes in ionic strength are sufficiently small during one Newton iteration so that the activity coefficients can be assumed to be constant.

According to one of these hypotheses, the Jacobian matrix can be easily computed by:

This is because:

Neglecting the derivative of the activity coefficient, one obtains:

Alternatively:

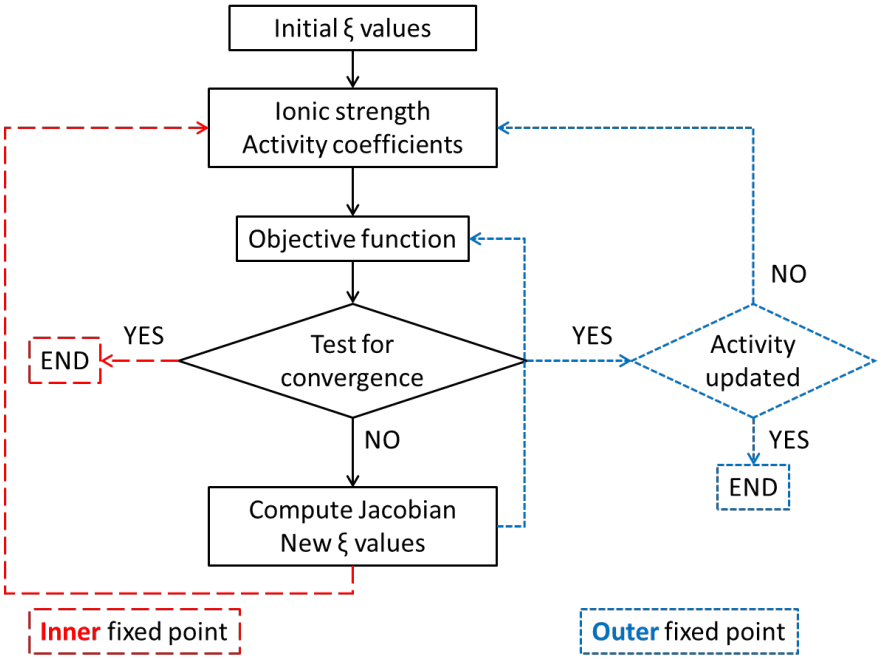

Figure 1: Inner and outer fixed-point activity algorithms

Inner fixed-point activity algorithm

This algorithm is the most commonly used algorithm. At each Newton loop, the activity coefficients are updated according to the new ionic force. From a strictly mathematical point of view, this method is then a quasi-Newton method because the Jacobian matrix Z is an approximation of the derivative of the objective function.

Outer fixed-point activity algorithm

Here, we propose this new algorithm. For a given set of values for the activity coefficients, the Newton procedure is iteratively run until convergence is achieved. Once convergence is reached, the activity coefficients are updated according to the ionic strength computed at the time of convergence. This procedure is repeated until no changes are detected in the activity coefficients. We then obtain a true Newton algorithm where the Jacobian matrix used is truly the derivative of the objective function.

Derivative of the activity coefficients in the Jacobian matrix

One can find an approach in the PHREEQC code [15] for including the derivative of the activity coefficients in the Jacobian matrix. By deriving the objective function, one obtains:

We then must calculate the derivative of the activity coefficients. Because the activity coefficients are strongly dependent on the ionic strength, we write:

We then compute two parts:

1. The derivative of the activity coefficients versus the ionic strength . This component is model-dependent and simple to compute regardless of which model is used. These elements are given for the 3 presented models in equations to .
2. The derivative of the ionic strength versus the logarithm of the activity components.

Computation method 1: a recursive formulation

Computing the derivative of the ionic strength yields:

We obtain a recursive formulation for the derivative of the activity coefficients:

This recursive formulation can be rearranged as:

The vector contains the explicit portion of the equation. This vector differs depending on the component used to derive the activity coefficient through the stoichiometric coefficient .

The matrix contains the coefficients of the implicit portion of the equation. This matrix is the same regardless of the component used in the derivation.

A complete derivative of the activity coefficients is given by the following linear system:

or

Computation method 2: an explicit formulation

Computing the derivative of the ionic strength yields:

We obtain an explicit formulation for the derivative of the ionic strength:

This allows for an explicit formulation of the derivative of the activity coefficients:

Derivative of the activity coefficients versus the ionic strength : Davis model

For a Davis model, the activity coefficients are given in logarithmic form by:

Taking the derivative of this equation yields:

Derivative of the activity coefficients versus the ionic strength : Extended Debye Hückel model

For an extended Debye Hückel model, the activity coefficients are given in logarithmic form by:

Taking the derivative of this equation yields:

Derivative of the activity coefficients versus the ionic strength : B-dot model

For the B-dot model, the activity coefficients are given in logarithmic form by:

Taking the derivative of this equation yields:

Test cases

Testing procedure

It is well-known that the initial guess of the values of the components plays a critical role in the convergence of the Newton methods [13, 21]. To test several initial guesses, we generate a large number (30 000) of activity component values according to the following procedure:

where and and are given in the description of the chemical test case to handle a representative range of concentrations.

We then obtain 30 000 realizations of the optimization procedure using the same chemical test case but different initial guesses. To analyze this large amount of data, we construct a frequency graph of the number of Newton iterations needed to reach convergence.

Chemical test cases

We propose 4 chemical test cases.

Test case with *only activity correction*

We first propose a test case without any chemical reactions. Any nonlinearity is only due to activity correction. The chemical system is composed of chloride ions, calcium ions, aluminum ions and thin ions. We neglect water dissociation and all chemical reactions. The details and equilibrium solutions are given in appendix 1.

Table 1: Chemical table for the test case with only activity correction.

	Cl^-	Ca^{2+}	Al^{3+}	Sn^{4+}	K
Cl^-	1	0	0	0	1
Ca^{2+}	0	1	0	0	1
Al^{3+}	0	0	1	0	1
Sn^{4+}	0	0	0	1	1
TOTAL	9.10^{-5}	10^{-5}	10^{-5}	10^{-5}	
Initial low I	5.10^{-7}	5.10^{-7}	5.10^{-7}	5.10^{-7}	
Initial high I	5.10^{-4}	5.10^{-4}	5.10^{-4}	5.10^{-4}	
	10^{-9}	10^{-9}	10^{-9}	10^{-9}	
	5.10^{-1}	5.10^{-1}	5.10^{-1}	5.10^{-1}	

One should note that this test case is not chemically realistic. Moreover, its numerical value comes only from activity correction if the unknowns of the nonlinear system are the logarithms of the activity components . Otherwise, if the unknowns are component concentrations , the problem becomes trivial and linear, and its solution is the total concentration .

Phosphoric acid test case

This test presents reactions between phosphoric acid and salt water. It includes 4 components and 8 chemical species. We handle only acid-base reactions: water dissociation and the 3 phosphoric acid reactions. A table including the stoichiometric coefficients, equilibrium constants, total concentrations and equilibrium solutions is given in appendix 2.

Gallic acid test case

This test case was proposed by Brassard and Bodurtha [20]. It includes 3 components and 17 chemical species. It is a classical test case, and many difficulties in convergence have been reported while solving it by using Newton or Newton-like algorithms [13, 14, 21]. A table including the stoichiometric coefficients, equilibrium constants, total concentrations and equilibrium solutions is given in appendix 3.

Iron-chromium test case

This test case concerns the rehabilitation of chromium-contaminated industrial soil using an iron-chromium reduction [2, 22]. Chromium (VI), which is the most toxic and mobile form of chromium, is reduced by iron (II) to yield chromium (III), which has a much lower solubility and is less toxic [23]. This test is reported to be a very difficult one [14, 21], so here we use some favorable testing conditions to increase the convergence of the Newton algorithm. A table including the stoichiometric coefficients, equilibrium constants, total concentrations and equilibrium solutions is given in appendix 4.

Results

Study of the test case with *only activity correction* during one resolution.

We first present two scenarios for the test case with *only activity correction* : one with a low ionic strength and one with a high ionic strength. The objective is to determine the influence of the activity correction on

the Newton procedure depending on the algorithm used. For the situation with a low ionic strength, this influence is expected to be negligible, whereas we expect a greater impact in the situation with a high ionic strength.

For the low ionic strength situation, the initial component activities are 5.0 10^{-7} mol. L^{-1} for all components. The ionic strength is 7.80 10^{-6} mol. $L^{-1,}$ and we obtain the species concentrations and activity values, which are given in *Table 2* . Also in *Table 2* , we show the first Newton steps proposed by the fixed-point algorithms (inner and outer) and by the full Newton algorithm.

Table 2: Initial values for the situation with a low ionic strength.

			Fixed-point	**Full Newton**
$\mathbf{Cl^-}$	5.02 10^{-7}	0.997	8.41	8.41
$\mathbf{Ca^{2+}}$	5.07 10^{-7}	0.987	19.00	18.58
$\mathbf{Al^{3+}}$	5.15 10^{-7}	0.970	19.00	18.06
$\mathbf{Sn^{4+}}$	5.27 10^{-7}	0.948	19.00	17.34

The Jacobian matrices given by both fixed-point algorithms are the same:

The full Newton algorithm leads to the following derivative matrix of the activity coefficients:

The following Jacobian matrix is also obtained:

For the scenario with a low ionic strength, the Jacobian matrices are equivalent for the full Newton and fixed-point algorithms, leading to similar Newton steps ().

The high ionic strength scenario starts with an initial component activities equal to 5.0 10^{-4} mol.$L^{-1}$for all components. The ionic strength equals 1.197 mol.L^{-1}. The species concentrations, activity values and first Newton steps proposed by the fixed-point algorithm (inner and outer) and full Newton algorithm are given in *Table 3* .

Table 3: Initial values for the situation with a high ionic strength.

			Fixed-point	**Full Newton**
$\mathbf{Cl^-}$	7.12 10^{-4}	0.702	-0.82	-1.07
$\mathbf{Ca^{2+}}$	2.06 10^{-3}	0.243	-0.98	-1.78
$\mathbf{Al^{3+}}$	1.20 10^{-2}	4.16 10^{-2}	-0.98	-2.75
$\mathbf{Sn^{4+}}$	1.42 10^{-1}	3.51 10^{-3}	-0.98	-4.12

The Jacobian matrices given by both fixed point algorithms are the same:

The full Newton algorithm leads to the following derivative matrix of the activity coefficients:

The following Jacobian matrix is also obtained:

For the scenario with a high ionic strength, the Jacobian matrices Z are very different for the full Newton and fixed-point algorithms, leading to very different Newton steps (). Moreover, we find an increase in the condition number of matrix Z for the full Newton algorithm. The condition number of matrix is 21.6, whereas it equals 1 for matrix .

The symmetry of the Jacobian matrix in equations and is specific to this test case. As shown in equation

, the Z matrix for the fixed-point algorithm is symmetric, whereas equation proves that it is usually not symmetric for the full Newton algorithm.

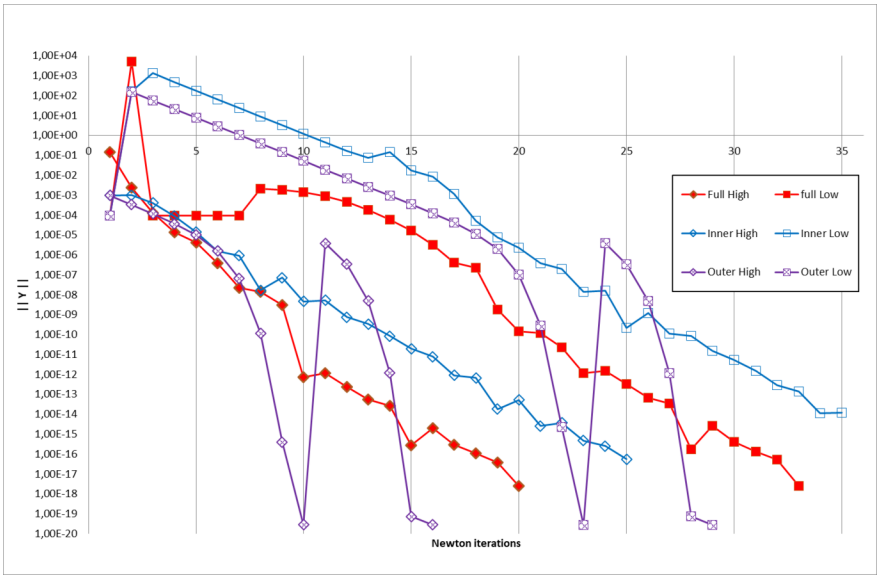

Figure 2 : Evolution of || Y || versus the number of Newton iterations for the test case with only activity correction

Comparing the three algorithms on the test case with *only activity correction* , one can see in Figure 2 that:

- The outer fixed-point algorithm requires the fewest Newton iterations to reach convergence, whereas the inner algorithm requires the most iterations. The full Newton algorithm requires an intermediate number of Newton iterations.
- For the tree algorithms, obtaining the solution at a low ionic strength requires less Newton iteration than at a high ionic strength. This point is obvious: for this test case, activity corrections are the only nonlinearity of the problem, and they are less important at a low ionic strength than at a high ionic strength.
- The outer fixed-point algorithm runs 3 minimization loops for the situations with both low and high ionic strength. The first loops converge at 10 (low) and 23 (high) iterations; the second loop converges at 15 (low) and 29 (high) iterations. The third loop is the*confirmation* loop used to check that no changes in the ionic strength computation occur and then to confirm the convergence of the algorithm.

Frequency graphs

We plot graphs of the cumulative ratio of the resolutions that converge within a given number of Newton iterations. According to the graph, the algorithm that reaches a cumulative frequency of 1 is said to be robust. The algorithm that reaches a high cumulative frequency for a low number of Newton iterations is said to be fast.

Test case with *only activity correction*

The test case with *only activity correction* is a simple chemical test case. It makes sense only for studying the activity correction algorithms. It is solved by all the algorithms (see Table 5) within 150 Newton iterations (Figure 3). The fastest algorithm is the outer fixed-point algorithm, regardless of the ionic strength. Moreover, this algorithm shows a very low sensitivity to the ionic strength by resolving the low ionic strength case within 24 or 25 iterations and the high ionic strength case within 21 iterations regardless of the initial

guess. The inner fixed-point and the full Newton algorithms are much more sensitive to the ionic strength, with significant increases in the number of iterations required to converge in the case with a high ionic strength. For this case, we find that the best algorithm is the outer fixed-point algorithm, and the inner fixed-point algorithm is the worst according to the number of Newton iterations. Taking the computing time of one Newton iteration into account (Table 4), we see that the full Newton algorithm is the slowest and the outer fixed-point algorithm is the fastest.

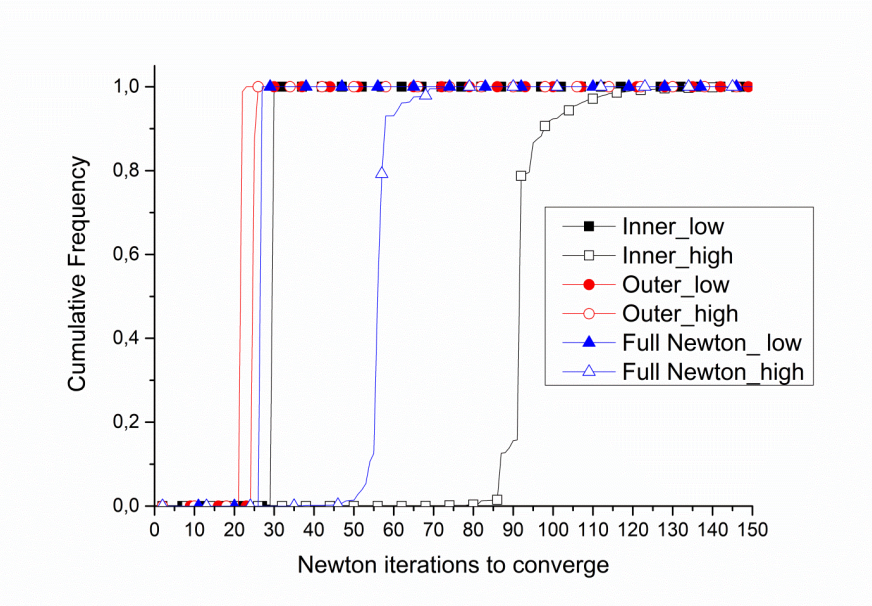

Figure 3: Frequency graph for the test case with only activity correction at low and high ionic strengths.

Phosphoric acid test case

This test case is a simple chemical test, which is interesting in this context because of the PO_4^{3-} species. We find that all the algorithms reach 100 % resolution (Table 5) in fewer than 70 iterations. The frequency graph shows that the full Newton algorithm usually converges within the fewest iterations and that the inner fixed-point algorithm requires the most iterations to converge. Nevertheless, the outer fixed-point algorithm provides very interesting results for this test case: it converges within 36 and 39 iterations regardless of the initial guess.

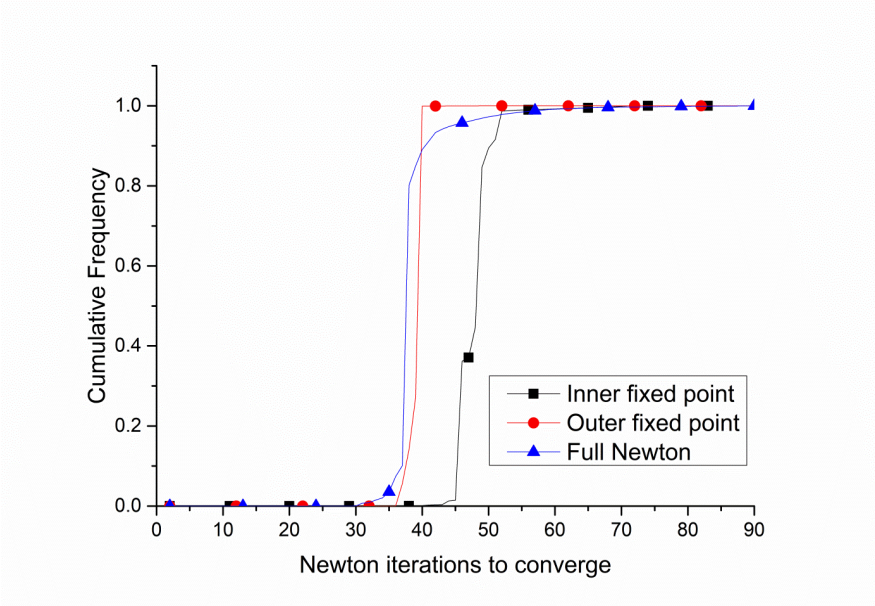

Figure 4: Frequency graph for the phosphoric acid test case

Gallic acid test case

The results for the gallic acid test case (Figure 5) confirm those of the phosphoric acid test case. The inner fixed-point algorithm requires the most iterations to converge (approximately 150). The full Newton algorithm usually requires fewer iterations (approximately 130) but sometimes requires many more iterations (200); sometimes it fails to converge at all (0.05 % failure rate, see Table 5). The outer fixed-point algorithm converges with the fewest iterations (77 iterations maximum). Moreover, this algorithm gives the sharpest frequency graph, indicating that it is not sensitive to the initial guess.

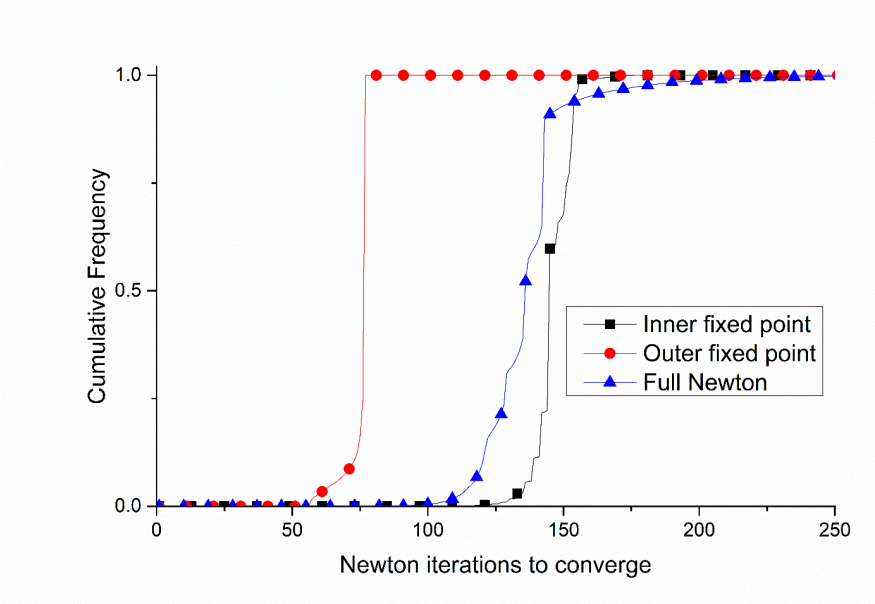

Figure 5: Frequency graph for the gallic acid test case

Iron-chromium test case

The iron-chromium test case is the strongest test used here. It is reported [14, 21] to generate very high condition numbers. The range of initial guesses is chosen with the goal of keeping this test reasonable. Nevertheless, one can see (Figure 6) that it is very hard for the full Newton algorithm to converge, with a 91.35 % failure rate (Table 5). Notably, this full Newton algorithm sometimes results in faster resolutions (according to the number of Newton iterations) because it is the only algorithm that sometimes converges with fewer than 200 Newton iterations. Both the inner and outer fixed-point algorithms converge most frequently within 230-240 iterations. The inner fixed-point algorithm reaches 80 % of its realizations after 240 iterations but needs up to 1000 iterations to complete the set and fails to converge at a rate of 1.62 % (Table 5). The outer fixed-point algorithm requires between 226 and 233 iterations to converge regardless of the initial point. Moreover, it always succeeds in solving this test case.

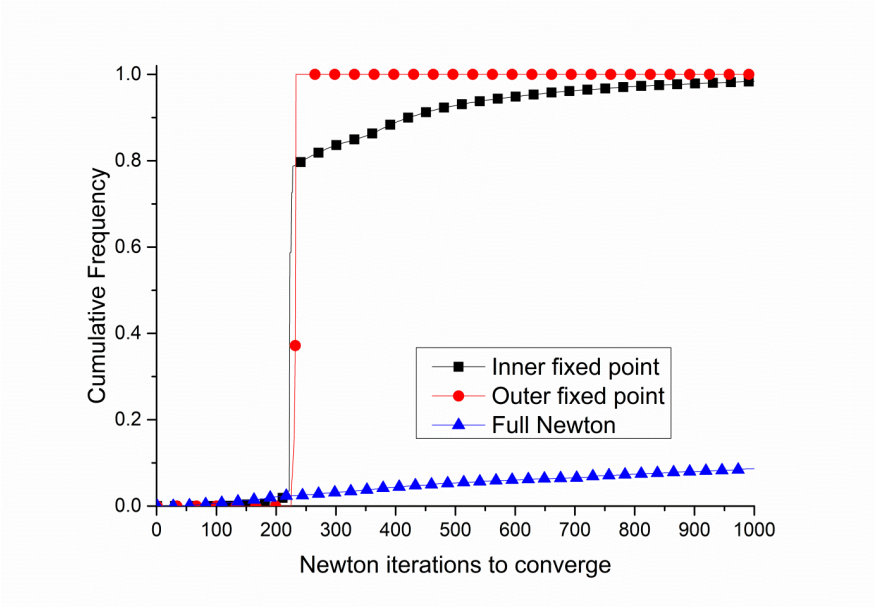

Figure 6: Frequency graph for the iron-chromium test case

Comparison of CPU times and failure ratios

As expected, the CPU times for one iteration loop increase as the number of species increases in the test case and as the complexity of the algorithm increases (see Table 4). We show that the outer fixed-point algorithm is the fastest and that the full Newton algorithm is the slowest. Moreover, the recursive formulation of the full Newton algorithm is the slowest. For this reason, we strongly recommend not using the recursive formulation, and we prefer the explicit formulation.

Table 4: CPU times for 1 loop (ms)

	Outer fixed-point	Inner fixed-point	Full explicit	Full recursive
Only I	$1.76\ 10^{-6}$	$2.18\ 10^{-6}$	$4.70\ 10^{-6}$	$1.74\ 10^{-4}$
Phosphoric acid	$2.34\ 10^{-6}$	$2.97\ 10^{-6}$	$4.72\ 10^{-6}$	$2.91\ 10^{-4}$
Gallic acid	$2.03\ 10^{-6}$	$2.68\ 10^{-6}$	$6.30\ 10^{-6}$	$6.23\ 10^{-4}$
Iron-chromium	$3.04\ 10^{-4}$	$1.15\ 10^{-5}$	$3.59\ 10^{-5}$	$1.29\ 10^{-3}$

We find that the only algorithm that solves all the test cases with a 100 % success rate is the outer fixed-point algorithm. The inner fixed-point algorithm is less robust and fails to solve the iron-chromium test 1.62 % of the time. The weakest algorithm is the full Newton algorithm, which fails at a 0.05 % rate for gallic acid test and at a 91.35 % rate for the iron-chromium test.

Table 5: Failure ratio after 1,000 iterations

	Only-I	Phosphoric acid	Gallic acid	Iron-Chromium
Inner	0.00 %	0.00 %	0.00 %	1.62 %
Outer	0.00 %	0.00 %	0.00 %	0.00 %
Full	0.00 %	0.00 %	0.05 %	91.35 %

Conclusion

We compared 3 algorithms based on their ability to handle activity correction in equilibrium chemistry solvers.

The full Newton algorithm is the most integrated algorithm from a mathematical point of view. Nevertheless, we found it to be the slowest and weakest algorithm. We suppose this algorithm increases the nonlinearity of the chemical system by injecting activity corrections into the mass action equations and conservation laws. It increases the condition number of the Jacobian matrix, as shown by comparing and . It has been shown [14, 21] that a condition number that is too high leads to inaccurate steps in the Newton methods, leading to numerical difficulties or nonconvergence. Because chemical equilibrium computation is still a highly nonlinear problem, increasing its nonlinearity by injecting activity correction seems to be an inefficient choice.

The inner fixed-point algorithm includes an intermediate integration of activity correction into the Newton loop. Both loops, Newton for the mass action equations and conservation laws and fixed-point for activity correction, run together. In this way, changes induced by activity correction disturb the Newton minimization. This point explains the convergence difficulties of the inner fixed-point algorithm when activity correction becomes important.

The outer fixed-point algorithm proposes a complete separation between the Newton and activity correction loops. In this way, nonlinearity induced by activity correction cannot disturb the Newton convergence, and the condition number of the Jacobian matrix is lower than that obtained by the full Newton algorithm. This leads to a more stable and robust algorithm. We found that the outer fixed-point algorithm is the fastest in terms of CPU times for one Newton iteration, usually faster than or equivalent to the other algorithms in terms of the number of required Newton iterations and the most robust.

According to the results presented here, we recommend the outer fixed-point algorithm. This algorithm is the least time consuming for one Newton iteration, it usually requires the fewest number of iterations, and it is the most robust and least sensitive to the initial guess. Moreover, its implementation with existing codes is very simple and requires very few modifications.

Acknowledgements

The authors acknowledge the French programme NEEDS for its financial support to the project NewSolChem

Bibliography

1. Lefèvre F, Sardin M, Schweich D (1993) Migration of strontium in clayey and calcareous sandy soil: Precipitation and ion exchange. J Contam Hydrol 13:215–229. https://doi.org/10.1016/0169-7722(93)90058-Z

2. Wanner C, Eggenberger U, Mäder U (2012) A chromate-contaminated site in southern Switzerland – Part 2: Reactive transport modeling to optimize remediation options. Appl Geochem 27:655–662. http://dx.doi.org/10.1016/j.apgeochem.2011.11.008

3. Werner AD, Bakker M, Post VEA, et al (2013) Seawater intrusion processes, investigation and management: Recent advances and future challenges. Adv Water Resour 51:3–26. http://dx.doi.org/10.1016/j.advwatres.2012.03.004

4. De Windt L, Burnol A, Montarnal P, van der Lee J (2003) Intercomparison of reactive transport models applied to UO2 oxidative dissolution and uranium migration. J Contam Hydrol 61:303–312

5. Wanner C, Druhan JL, Amos RT, et al (2015) Benchmarking the simulation of Cr isotope fractionation. Comput Geosci 19:497–521. https://doi.org/10.1007/s10596-014-9436-9

6. Steefel CI, Appelo CAJ, Arora B, et al (2015) Reactive transport codes for subsurface environmental simulation. Comput Geosci 19:445–478. https://doi.org/10.1007/s10596-014-9443-x

7. Yeh GT, Tripathi VS (1989) A critical evaluation of recent developments in hydrogeochemical transport models of reactive multichemical components. Water ResourRes **25** :93–108

8. Pruess K, Garcia J, Kovscek T, et al (2004) Code intercomparison builds confidence in numerical simulation models for geologic disposal of CO2. Energy 29:1431–1444

9. Yaron B, Dror I, Berkowitz B (2010) Contaminant geochemistry-a new perspective. Naturwissenschaften 97:1–17

10. Steefel CI (2000) New directions in hydrogeochemical transport modeling: Incorporating multiple kinetic and equilibrium reaction pathways. Comput Methods Water Resour 1:331–338

11. van der Lee J, De Windt L (2001) Present state and future directions of modeling of geochemistry in hydrogeological systems. J Contam Hydrol 47:265–282

12. Steefel CI, DePaolo DJ, Lichtner PC (2005) Reactive transport modeling: An essential tool and a new research approach for the Earth sciences. Earth Planet Sci Lett 240:539–558

13. Carrayrou J, Mosé R, Behra P (2002) New efficient algorithm for solving thermodynamic chemistry. AIChE J 48:894–904

14. Marinoni M, Carrayrou J, Lucas Y, Ackerer P (2017) Thermodynamic equilibrium solutions through a modified Newton Raphson method. AIChE J 63:. https://doi.org/10.1002/aic.15506

15. Parkhurst DL, Appelo CAJ (1999) User's guide to PHREEQC (version 2)- A computer program for speciation, batch-reaction, one-dimensional transport, and inverse geochemical calculations. *Denver, CO, USA.*

16. Steefel C (2006) Crunch – user's guide. USA: Lawrence Berkeley Laboratory

17. Van der Lee J (1998) Thermodynamic and mathematical concept of CHESS. Ecole des Mines de Paris

18. Westall JC, Zachary JL, Morel FMM (1976) MINEQL: a computer program for the calculation of chemical equilibrium composition of aqueous system. Cambridge

19. Morel F, Morgan J (1972) Numerical method for computing equilibriums in aqueous chemical systems. Environ Sci Technol 6:58–67. https://doi.org/10.1021/es60060a006

20. Brassard P, Bodurtha P (2000) A feasible set for chemical speciation problems. Comput Geosci 26:277–291

21. Machat H, Carrayrou J (2017) Comparison of linear solvers for equilibrium geochemistry computations. Comput Geosci 21:131–150. https://doi.org/10.1007/s10596-016-9600-5

22. Wanner C, Eggenberger U, Kurz D, et al (2012) A chromate-contaminated site in southern Switzerland – Part 1: Site characterization and the use of Cr isotopes to delineate fate and transport. Appl Geochem 27:644–654. http://dx.doi.org/10.1016/j.apgeochem.2011.11.009

23. Richard FC, Bourg ACM (1991) Aqueous geochemistry of chromium: A review. Water Res 25:807–816. https://doi.org/10.1016/0043-1354(91)90160-R