PyEquIon: A Python Package For Automatic Speciation Calculations of Aqueous Electrolyte Solutions

5 authors, including:

Caio Marcellos
Federal University of Rio de Janeiro
14 PUBLICATIONS **8** CITATIONS

Elvis Do Amaral Soares
Federal University of Rio de Janeiro
12 PUBLICATIONS **16** CITATIONS

Fabio Ramos
Federal University of Rio de Janeiro
19 PUBLICATIONS **197** CITATIONS

Amaro Barreto Jr
Federal University of Rio de Janeiro
80 PUBLICATIONS **319** CITATIONS

Some of the authors of this publication are also working on these related projects:

Project   Undergraduation final project View project

Project   Protein adsorption View project

PyEquIon: A Python Package For Automatic Speciation Calculations of Aqueous Electrolyte Solutions

Caio Felippe Curitiba Marcellos[a,], Gerson Francisco da Silva Junior[a], Elvis do Amaral Soares[a], Fabio Ramos[b], Amaro G. Barreto Jr[a]

[a]*Department of Chemical Engineering, Center of Technology, Federal University of Rio de Janeiro, Rio de Janeiro, 21941-909, RJ, Brazil*
[b]*Department of Applied Mathematics, Institute of Mathematics, Federal University of Rio de Janeiro, Brazil*

Abstract

In several industrial applications, such as crystallization, pollution control, and flow assurance, an accurate understanding of the aqueous electrolyte solutions is crucial. Electrolyte equilibrium calculation contributes with the design and optimization of processes by providing important information, such as species concentration, solution pH and potential for solid formation. In this work, a pure Python library distributed under BSD-3 license was developed for the calculation of aqueous electrolyte equilibrium. The package takes as inputs the feed components of a given solution, and it automatically identifies its composing ions and the chemical reactions involved to calculate equilibrium conditions. Moreover, there is no established electrolyte activity coefficient model for a broad range of operational conditions. Hence, in this package, built-in activity coefficient models are structured in a modular approach, so that the non-ideality calculation can be performed by a user provided function, which allows further research in the topic. The package can be used by researchers to readily identify the equilibrium reactions and possible solid phases in a user friendly language.

Keywords: electrolyte equilibria, scientific computing, chemical speciation

---

[*]Corresponding author.
E-mail address: caiocuritiba@gmail.com

1. Introduction

Aqueous electrolyte solutions are of major interest in many industrial and environmental applications and comprise different research fields, such as geological, pharmaceutical, biological, chemical, nuclear and petroleum engineering. Equilibrium calculations of aqueous electrolytes systems are often implemented in mathematical models for designing and monitoring of processes [1, 2]. However, despite the widespread literature on electrolyte systems, the theory of electrolyte thermodynamics is not completely understood with a lack of consensus in model formulations [3]. Some of the main challenges in the thermodynamic modeling of electrolyte solutions are [4]: (i) the choice of computer code and appropriate parameter database; (ii) proper implementation of published models, because the informed formulation can be often inadequate; and (iii) an inappropriate emphasis of studies in modifications and extension of models without substantial physical foundation.

There are reliable open-source and commercial computational tools for calculating chemical equilibrium of aqueous solution with a comprehensive set of compounds and thermodynamic database, [5, 6, 7, 8, 9]. Some of them were developed few decades ago in technologies that are now outdated,

so that it an be cumbersome to set the computational environment for its proper usage. Additionally, for some packages, it can be difficult for a non-programmer to access the code and to modify it for a specific need. For non-expert researchers or industrial analysts, chemical electrolyte packages are often used in a black-box manner, which can make studies less reproducible and error prone.

General-purpose open source computing languages, such as Python, have garnered great attention from the scientific community due to dynamic prototyping and high-level data structure, which allows for both sophisticated development and scripting [10]. As a multipurpose language, it enables interactions between services enhancing the overall research workflow, such as data manipulation, model creation, and data visualization. Moreover, it provides good code readability, interoperability with other languages, a built-in documentation system, rich collection of data structures and a variety of libraries for scientific computing [11, 12].

In this work, we present a computational tool for the aqueous speciation calculation with automatic equilibrium reactions determination. Given a set of input components, the software enables the automatic ions identification and the involved equilibrium reactions. The final equation system is formed by the electroneutrality equation, mass balances and, if necessary, a closing equation. For the efficient usage in iterative numerical methods, the package offers just in time compilation, automatic jacobian generation and code exportation of a residual function for incorporation in differential algebraic equation systems. A modular approach for nonideality calculation allows the incorporation of user-provided functions for the species activity coefficient. This tool is a contribution for the researchers' understanding of electrolyte chemical reactions, and it can also be used for numerical simulation.

2. Aqueous Electrolyte Equilibrium Overview

A mixture of solutes in a solvent can originate various species, such as free ions, undissociated molecules, ion pairs, complexes or chelates, and micelle cluster [13]. Thus, it is necessary to identify all possible species present in the solution, and all the equilibrium reactions between them.

The fundamental law of mass action [14] states that for a given reaction

$$a\,\mathrm{A} + b\,\mathrm{B} \rightleftharpoons c\,\mathrm{C} + d\,\mathrm{D}, \tag{1}$$

3
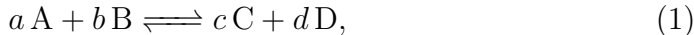
the reaction equilibrium constant K can be defined as:

$$K = \frac{\{C\}^c \{D\}^d}{\{A\}^a \{B\}^b},$$

(2)

with the bracket operator denoting activity.

The activity of a compound $\{A\}$ is the effective concentration of A in the solution given by $\{A\} = \gamma_A [A]$, where $[A]$ is the molal concentration of A and γ_A is its activity coefficient. The activity coefficient describes the deviation from the ideal solution, and the nonideality is primarily due to electrostatic interactions between the ions. It can be observed experimentally, for instance, if one computes the equilibrium constant directly with ions concentration for various ionic strength the equilibrium constant may vary. Another observation is the variation of molar conductivities of strong electrolytes within a concentration range [13].

A common approach for calculating activity coefficients is to use the Debye-Hückel (DH) theory, which was derived from the Poisson-Boltzmann equation considering as the source of nonideality the electrostatic double-layer interaction [15]. The Debye-Hckel limiting law, for very dilute solutions, relates the activity coefficient with the ionic strength (Eq. 3) and is given by Eq. 4:

$$I = \sum_{i \in C} [i] \, z_i{}^2,$$

(3)

$$\log \gamma_A = -A_\phi z_i^2 \sqrt{I},$$

(4)

where z_i is the charge of specie, C is the set of all species and A_ϕ a temperature dependent parameter given by Eq. 5:

$$A = \left(\frac{e^2}{\epsilon_0 \, \epsilon_r \mathrm{RT}} \right)^{\frac{3}{2}} \frac{N_A}{8\pi} \sqrt{2\rho_w}$$

(5)

with e as the electron charge, ϵ_0 and ϵ_r are the vacuum and relative permittivity, respectively, ρ_w is the solvent density, N_A is the Avogadro's constant and T is the temperature [3].

The DH model is only accurate for very dilute solutions because it does not take into account some factors, for example, short-range interaction forces, which are very important for more concentrated solutions. For concentrated solution the main approaches are [3]: (i) Physical models: are

derivation from the DH model in which the most cited is the Pitzer model and consider interactions due to excluded volume and Van der Waals attraction; (ii) Chemical models: the nonideality is attributed to semi-stable chemical species (solvated ions), for instance, the Robinson and Stokes model and (iii) Local-composition models: the physical van der Waals interaction are considered as a function of local composition, such as NRTL, Wilson or UNIQUAC models. All those approaches require the fitting of parameters from experimental data.

From the nonideality models derived from DH equation, there are [16, 17, 18]: (i) the extended DH (1923), (ii) the Guggenheim Model (1935) with includes two terms for ion interaction and is a precursor of the Pitzer model; (iii) the Davies model (1938) with received attention since has no ion-specific parameter; (iv) Hückel Equation (or B-dot) that is used in this work and has two ion-specific parameters; (v) a polynomial DH model by the National Bureau of Standards [19] for higher ionic strength; (vi) corrected DH (1983) with the account for finite ion size; (vii) the Bromley [20] activity model that can model multicomponent system of I up to 6 molal and (viii) the Pitzer Equation which is a semi-empirical extended DH with virial expansion and requires high-order interaction parameters for high concentrated solutions.

A fully experimental nonideality method is the Meissner Reduced Activity Coefficient Model [21] with regression for a modified activity variable related to the ionic strength. Local-composition models are, for instance, the Multicomponent e-UNIQUAC [22] and the e-NRTL (Cardoso et. al., 1986). Recently, the Generalized DH model [23] was proposed based on Poisson-Fermi equation with the goal of describing the activity coefficient with fewer parameters, although it still requires more validation for multicomponent systems.

The Davies equation, Eq. 6, is often used for low concentration solutions (up to 0.5 M). The B-dot equation, Eq. 7, is a virial-type expansion for the Debye-Hückel equation where b is an adjustable parameter, which is the difference between the observed activity coefficient logarithm and the calculated activity coefficient logarithm by the Debye-Hückel equation. With this approach, this model becomes accurate at concentrations up to 1 M [16].

$$\log \gamma_i = -Az_i^2 \frac{\sqrt{I}}{1 + \sqrt{I}} - 0.3I. \tag{6}$$

5

$$\log \gamma_i = -\frac{z_i^2 A \sqrt{I}}{1 + B a_i \sqrt{I}} + b_i I. \tag{7}$$

For more concentrated solutions, the most used model is the ion-interaction model proposed by Pitzer [24]. This model takes into account long and short-range interactions, where long-range forces are presented by the extended DH model, and short-range forces are presented as a virial type expansion with parameters of interaction between two or three particles of the solute [3, 16, 17, 25, 26]. The parametric expressions used by Pabalan and Pitzer [25] are presented in Eq. 8 and in Eq. 9 for an M cation and an X anion of interest in a mixture of electrolytes.

$$\begin{aligned} \ln \gamma_M = z_M^2 F + \sum_a m_{a'}(2B_{Ma} + ZC_{Ma} + \sum_c m_c(2\Phi_{Mc} + \sum_a m_a \psi_{Mca})) \\ + \sum_a \sum_{a'} m_a m_{a'} \Psi_{Maa'} + |z_M| \sum_c \sum_a m_c m_a C_{ca} + \sum_n m_n(2\lambda_{nM}) \end{aligned} \tag{8}$$

$$\begin{aligned} \ln \gamma_X = z_X^2 F + \sum_c m_c(2B_{cX} + ZC_{cX} + \sum_a m_a(2\Phi_{Xa} + \sum_c m_c \psi_{Xac})) \\ + \sum_c \sum_{c'} m_c m_{c'} \Psi_{Xcc'} + |z_X| \sum_c \sum_a m_c m_a C_{ca} + \sum_n m_n(2\lambda_{nX}) \end{aligned} \tag{9}$$

Where F account the long-range interaction forces and its value is calculated according to Eq. 10 and Z is defined by Eq. 11, c and c' are cations and a and a' are anions, m_c and m_a are the concentrations on the cation and anion molal scale, respectively.

$$\begin{aligned} F = A_\phi \left[\frac{\sqrt{I}}{1 + b\sqrt{I}} + \frac{2}{b} \ln(1 + b\sqrt{I}) \right] + \sum_c \sum_a m_c m_a B'_{ca} + \\ \sum_c \sum_{c'} m_c m_{c'} \Phi'_{cc'} + \sum_a \sum_{a'} m_a m_{a'} \Phi'_{aa'} \end{aligned} \tag{10}$$

$$Z = \sum_i m_i |z_i| \tag{11}$$

The B and B' parameters are calculated by Eq. 12 and 13. The parameters β are determined empirically from binary systems. The variable B' is the ionic strength derivatives of B. The C term is determined by Eq. 16, where C^ϕ parameter is also determined empirically from binary data.

$$B_{MX} = \beta_{MX}^{(0)} + \beta_{MX}^{(1)} g(\alpha\sqrt{I}) + \beta_{MX}^{(2)} g(\alpha\sqrt{I}) \tag{12}$$

$$B' = \frac{1}{I}(\beta_{MX}^{(1)} g'(\alpha\sqrt{I}) + \beta_{MX}^{(2)} g'(\alpha\sqrt{I})) \tag{13}$$

$$g(x) = \frac{2}{x^2}[1 - (1+x)\exp(-x)] \tag{14}$$

$$g'(x) = \frac{-2}{x^2}[1 - (1 + x + \frac{x^2}{2})\exp(-x)] \tag{15}$$

$$C = \frac{C_{MX}^{\phi}}{2\sqrt{|z_i z_j|}} \tag{16}$$

On the other hand, the parameter ψ can be obtained from experimental data of mixed electrolytes and the parameter Φ and Φ' are determined as described in Pitzer (1975) [26].

3. Software description

Pyequion is a free python library distributed under BSD license for the automatic determination of equilibrium reaction and species. In short, for a user provided component mixture, reactions as in Eq. 1 are determined together with the involved species. Reaction constants are obtained from a reaction database and the activity coefficients are calculated from a selected model with species parameters from a database. The nonlinear model can be exported for embedding in dynamic simulations.

A representational scheme of the internal steps for equilibrium calculation is illustrated in Figure 1. The identification of the reactions involved is the main part of the computational tool. A database file containing various ionic equilibrium reactions in the JSON (Javascript Object Notation) or python dictionary format was created from the database file distributed with PHREEQC [7]. When all species in one side of an equilibrium reaction are known, the species on the other side can be formed. Thus, they are added to the system. Hence, by a recursive procedure, all reactions are traversed, checking for the condition of all species known in one side of the equilibrium reaction. The recursive algorithm is applied to all the new compounds added to the system.
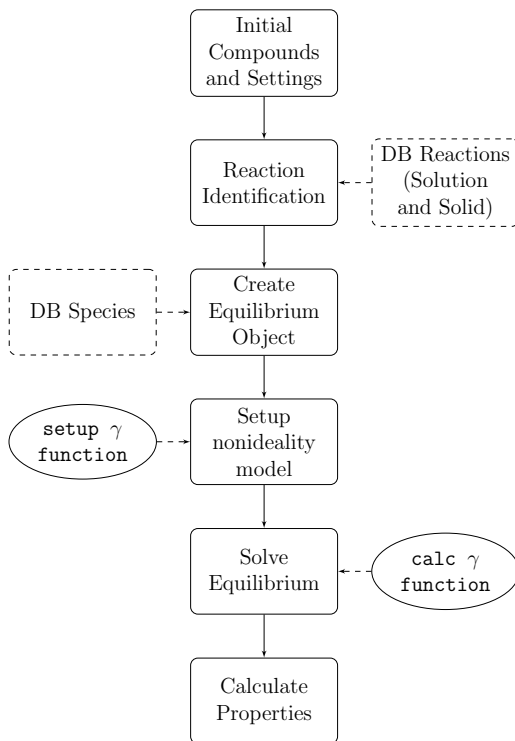
Figure 1: Illustrative scheme for the required steps in a equilibrium calculation.

At the end of the reaction identification procedure, the reactions and the present species are identified. The species are further processed to get information from its textual representation: its charge and the composing elements with their coefficients, by recognizing that each new element starts with an uppercase letter. This approach avoids redundancies in the species database.

Each chemical compound is related to another JSON (or python *dict* datatype) database file of species with the required information for the nonideality model. The database file contains ion specific and interaction parameters that can be used by the chosen nonideality model. In this software version, the following models and companion parameters files are provided: (i) extended Debye-Hückel equation or *B-dot* [27, 28]; (ii) Bromley method with individual ions parameter [20]; (iii) SIT formula [29]; (iv) Pitzer method [24]. The parameters for the species activity coefficient calculation were obtained from the PHREEQC database files for *B-dot*, SIT and Pitzer after conversion to the universally known structured file format (JSON). The pa-

rameters for the Bromley method were obtained from the original paper [20] using the individual ion correlation. The parameters files are not exhaustive and interaction parameters may be missing. Users can provide new thermodynamic models using a setup function, which relates the species to the database, and the calculation function for the decimal logarithm of the species activity coefficient.

3.1. Software Functionalities

The following list enumerates the main functionalities provided by *pyequion* v0.0.5.

- Pure python package, without compiled dependencies, making it easy to install by non skilled programmers;

- Automatic identification of species and reactions for a given input compound list;

- Suitable integration with *jupyter* notebooks for latex formatted display of reactions, which allows students and researchers to improve their understanding of the system;

- Built-in activity coefficient models: *B-dot*, *Bromley*, *SIT* and *Pitzer;*

- Modular approach for user defined functions for the calculation of activity coefficients;

- Reactions and parameters database already in convenient data interchange format (JSON);

- The `SolutionResult` class provide properties of the solution equilibrium, as: pH, conductivity, ionic strength, concentrations and activities, dissolved inorganic carbon, solid phases saturation index, ionic activity product or the concentrations in case of equilibrium with a solid phase and the reactions used;

- Solid phase equilibrium to calculate precipitated concentration and solubilities;

- Option for just-in-time compilation of the nonlinear residual calculation using the package `numba` [30];

- Code generation to export the residual calculation as a python function, hence allowing the calculation to be used in environments without the package and to be coupled in phenomenological models, such as in differential algebraic equations;

- Jacobian generation to a source file with the aid of symbolic computation (`sympy` [31]), which can also be just-in-time compiled;

- Henry law for CO_2 vapour-liquid equilibrium with ideal or with the Peng-Robinson equation of state;

- Tutorial with relevant use cases.

3.2. Software validation

The software was validated against the output of the reference software PHREEQC [7] for low solute concentrations and in atmospheric conditions using the `pytest` framework for automatic unit tests. The software is accompanied by sample code comparing against experimental data for mean activity coefficients. The user should be aware that electrolyte equilibrium models are not established in the literature [32, 33, 4] and the results should be analyzed with caution, specially for modifications in conditions of temperature, pressure and concentrations.

4. Illustrative Examples

4.1. Basic usage

Figure 2 depicts the *pyequion* usage for aqueous equilibrium for the mixture of $NaHCO_3$ and $CaCl_2$. The compositions are specified from the molarity of each component in millimolar and the closed system solution is considered by default. In this case, the calculated pH is 7.87 and the system is supersaturated for the calcite calcium carbonate polymorph, but not for halite.

Figure 2: Example of equilibrium calculation for $NaHCO_3$ and $CaCl_2$ system using *pyequion* showing the involved reactions.

There are other options available that are described in the package documentation and exemplified in Listing 1. The example shows the solution for the same mixture modifying arguments as: (i) open system, allowing the equilibrium with calcite solid phase; (ii) selecting Peng-Robinson model for the fugacity coefficient in the vapour phase; (iii) adjusting the CO_2 partial pressure for the atmospheric (default); and (iv) using one of the available activity coefficient model, the Pitzer method. Besides listing the involved reactions and species, the software can generate a graph of the reactions, which provides intuition on the species' connections, as shown in Fig. 3.

```
1   import pyequion
2   sol = pyequion.solve_solution(
3       {'NaHCO3': 50, 'CaCl2': 10},
4       allow_precipitation=True,
5       close_type=pyequion.ClosingEquationType.OPEN,
6       activity_model_type='pitzer',
7       solid_equilibrium_phases=['Calcite'],
8       co2_partial_pressure=pyequion.pCO2_ref,
9       fugacity_calculation='pr',
10  )
11  \label{list_use1}
```

Listing 1: Example of using *PyEquIon* for aqueous equilibrium calculation modifying calculation options.
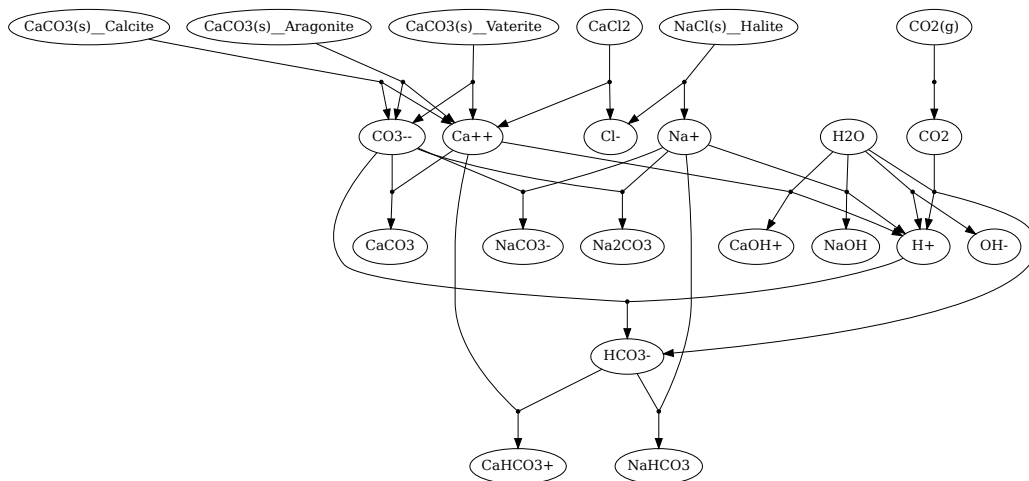
Figure 3: Graph of reactions for the $NaHCO_3$ and $CaCl_2$ open system with solid formation.

4.2. Calculating Mean Activity Coefficient with Pitzer Model

In this case study, the aqueous electrolyte equilibrium of NaCl, $CaCl_2$ and $MgCl_2$ are separately calculated using the Debye-Hückel (*B-dot*), SIT, Bromley with individual ion parameters and the Pitzer ion association method. The setup and calculation functions are provided by the package for the models and can be selected as presented in Listing 2.

```
sys_eq = pyequion.create_equilibrium(
    feed_compounds=['CaCl2'],
    initial_feed_mass_balance=['Cl-']
)

cIn_span = np.geomspace(1e-3, 3.0, 61)
methods = ['debye', 'bromley', 'sit', 'pitzer']

solutions = {
    key: [
            pyequion.solve_solution( {'CaCl2': c*1e3}, sys_eq
    ,
            activity_model_type=key)
            for c in cIn_span
        ]
    for key in methods
}
gamma_means = {key: np.array(
    [pyequion.get_mean_activity_coeff(sol, 'CaCl2')
```

```
19      for sol in solutions[key]
20      ])
21      for key in methods
22  }
```

Listing 2: Calculating mean activity coefficient for CaCl$_2$ with different thermodynamic models.

Figure 4 shows the comparison of the calculated mean activity coefficient with the different methods for NaCl, CaCl$_2$ and MgCl$_2$ electrolytes with experimental data [34, 35]. The aqueous equilibrium calculation is performed and the mean activity coefficient is provided by the package using the `get_mean_activity_coeff` function. A good agreement with the experimental data is obtained when using the Pitzer model. The other models start to deviate from the experimental data for higher ionic strength. In this example, the equilibrium object is created beforehand and it is used multiple times in the calculations.
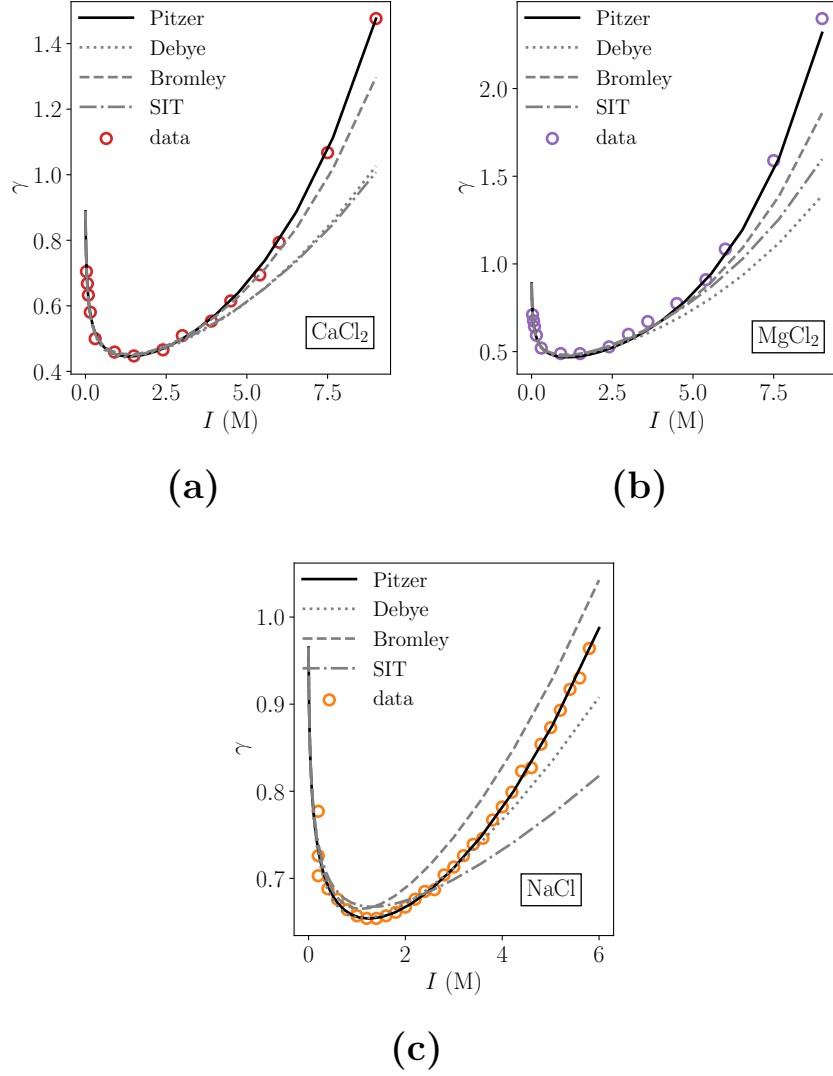
Figure 4: Comparison of mean activity coefficient of $CaCl_2$, $MgCl_2$ and NaCl using Debye-Hückel (*B-dot*), Bromley, SIT and Pitzer models.

4.3. User provided function for nonideality model

One of the main goals of this software package is to enable easy prototyping and exploration of nonideality models. In this section, we exemplify the usage of a custom function for performing activity coefficient calculation with the Bromley method [16, 20] using salt specific parameters.

For setting a custom function for $\log \gamma$ calculation, the function `solve_-solution` accepts a setup and a calculation function (Listing 3). The former is used for collecting species parameters for the model and for any preprocessing, and it is executed before the nonlinear equation system. The latter is for actually calculating $\log \gamma$ for each species during the residual evaluation of the nonlinear system.

```
1  sys_eq = pyequion.create_equilibrium(['KCl'])
2  solution = pyequion.solve_solution({'KCl': 3e3}, sys_eq,
3      setup_log_gamma_func=setup_bromley_method,
4      calc_log_gamma=calc_bromley_method
5  )
6  gamma_mean = pyequion.get_mean_activity_coeff(solution, 'KCl'
      )
```

Listing 3: Code sample for passing a custom activity coefficient model applied to KCl.

An example of a setup function for the Bromley model is depicted in Listing 4. The model parameters can be obtained by a database file (such as JSON), but in this example, a global scoped hash table data structure is used, which contains parameters for NaCl and KCl. In line 1 and 2, the anions and cations are obtained, then each cation is associated with an anion parameter providing the B_{ij} Bromley's interaction parameter.

```
1  bromleyDB = {
2      'K+': {'Cl-': 0.024},
3      'Na+': {'Cl-': 0.0574},
4  }
5  def setup_bromley_method(reaction_sys, T, db_species, c_feed)
      :
6      anions = [sp for sp in reaction_sys.species if sp.z < 0]
7      cations = [sp for sp in reaction_sys.species if sp.z > 0]
8      for c in cations:
9          for a in anions:
10             try:
11                 c.p_scalar[a.name] = bromleyDB[c.name][a.name
      ]
12             except KeyError:
13                 c.p_scalar[a.name] = 0.0
14     return
```

Listing 4: Code sample for a custom activity coefficient model applied to KCl.

A calculation function is presented in Listing 5. Again, the anions and cations are obtained from the list of species. The calculation is performed

for each cation and anion, accordingly with the Bromley's method using parameters for the salt (not from individual ions). The function `bromley_-model_ion` is not shown here, but it calculates the $\log\gamma_i$ of an ion i giving the ionic strength I, interaction parameters $B_{i,j}$, species charge z_i and z_j and the molalities m_j.

```python
def calc_bromley_method(idx_ctrl, species, I, T):
    anions = [sp for sp in species if sp.z < 0]
    cations = [sp for sp in species if sp.z > 0]

    for c in cations:
        Bi_j = np.array([c.p_scalar[a.name] for a in anions])
        z_j = np.array([a.z for a in anions])
        mj = np.power(10, np.array([a.logc for a in anions]))
        loggC = bromley_model_ion(I, Bi_j, c.z, z_j, mj)
        c.set_log_gamma(loggC)

    for a in anions:
        Bi_j = np.array([c.p_scalar[a.name] for c in cations
    ])
        z_j = np.array([c.z for c in cations])
        mj = np.power(10, np.array([c.logc for c in cations])
    )
        loggA = bromley_model_ion(I, Bi_j, a.z, z_j, mj)
        a.set_log_gamma(loggA)
```

Listing 5: Code sample for a passing a custom activity coefficient model applied to KCl.

The Bromley's method was used for comparison with experimental data for a single electrolyte mean activity coefficient for NaCl and KCl. Furthermore, this model was also used for the solubility calculation of KCl in NaCl aqueous solution. Those results are summarized in Figure 5.
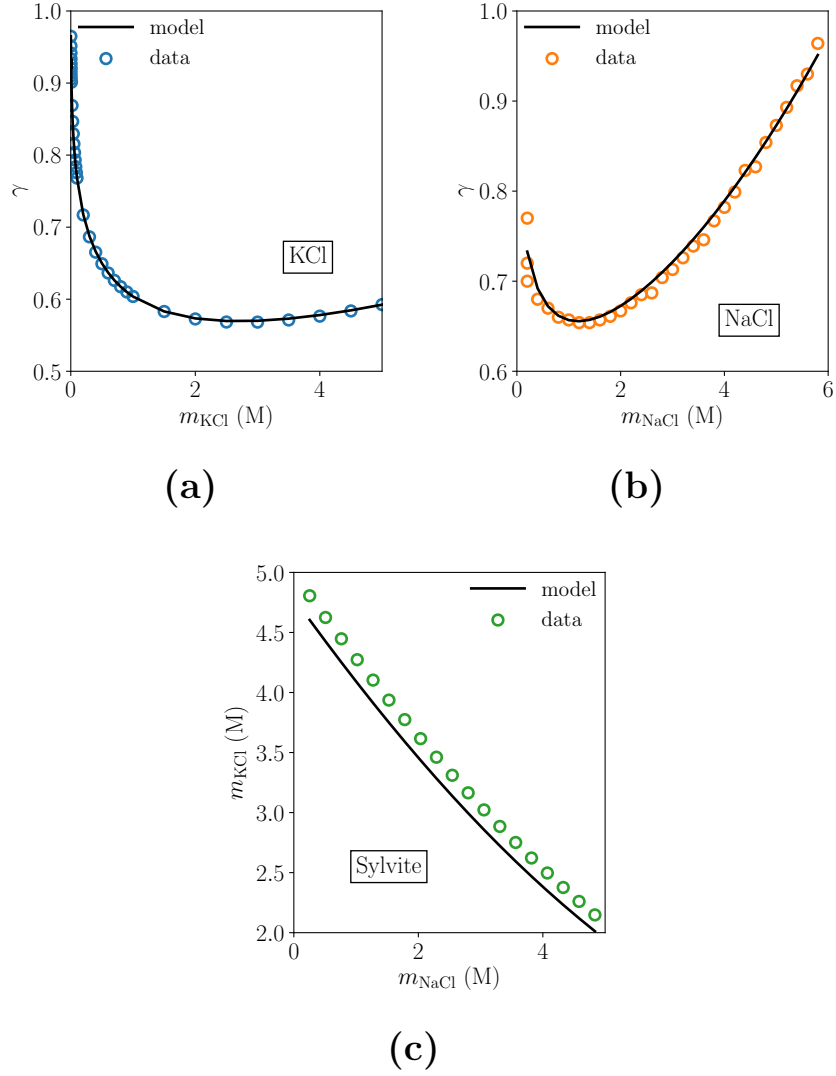
(a)

(b)

(c)

Figure 5: Using *pyequion* with a user provided nonideality model (Bromley's method with salt specific parameter) for (a) γ_{KCl} and (b) γ_{NaCl} single electrolyte solutions and (c) for the solubility of KCl(s) (Sylvite) in NaCl solution.

4.4. Embedding equilibrium calculation in dynamic simulation of crystallization

As an illustrative example on the usage of the provided package in a simulation of a dynamic system, we show the use of *pyequion* in the modelling of

crystallization of calcium carbonate by the mixture of $NaHCO_3$ with $CaCl_2$. The case study considers the unseeded crystallization of calcite by mixing 15 mM of $NaHCO_3$ with 5 mM $CaCl_2$ at a temperature of 298 K. The crystallization model consists of a population balance solved by the Standard Method of Moments [36], which reduces the partial differential equation to a set of ordinary differential equations, and by the mass balances of total dissolved calcium and total dissolved inorganic carbon. The kinetic parameters were taken from [37] and are reproduced in Table 1. The differential equations are presented by Eq. 17, where S is the supersaturation, defined as $\left(\frac{\{Ca^{2+}\}\{CO_3{}^{2-}\}}{K_{sp}} \right)^{(1/2)}$ with K_{sp} as the calcite solubility constant; [Ca] and [C], molal concentration of element Ca and C; $\rho_{c,mol}$, crystal molar density; k_v, shape factor; μ_2, second order moment of the particle size distribution; k_G, growth rate constant; g, growth rate exponent; k_B, nucleation rate constant; E_B, activation energy for nucleation rate.

Table 1: Parameters for the crystallization study using *pyequion* for supersaturation calculation.

Variable	Value	Unit
ρ_c	2.71×10^3	kg/m^3
k_v	1.0	—
E_B	12.8	—
k_B	1.4×10^{18}	$\#/m^3\,s$
g	1.0	—
k_G	5.6×10^{-10}	m/s

$$\frac{d[Ca]}{dt} = -3\rho_{c,mol}k_v\mu_2 \left[k_G(S-1)^g \right] \tag{17a}$$

$$\frac{d[C]}{dt} = -3\rho_{c,mol}k_v\mu_2 \left[k_G(S-1)^g \right] \tag{17b}$$

$$\frac{d\mu_j}{dt} = \begin{cases} k_B S \exp\left[-\dfrac{E_B}{(\log S)^2} \right] & \text{if } j = 1 \\ j\left[k_G(S-1)^g \right]\mu_{j-1} & \text{if } 1 < j \le 4 \end{cases} \tag{17c}$$

The equilibrium is calculated from an exported function using *pyequion*, as exemplified in Listing 6. The equilibrium is created using the individuals ions instead of the compounds $NaHCO_3$ and $CaCl_2$ allowing for independent definitions of each element. The exported function is embedded in the differential algebraic equation framework `daetools` [38], hence the equilibrium calculation residuals are algebraic constraints in the model. Figure 6 presents the profiles of pH, supersaturation, suspended particles mass concentration ($c_{\text{cryst}}=\mu_3 k_v \rho_c$) and mean particle size ($d_{10} = \mu_1/\mu_0$).

```python
sys_eq = pyequion.create_equilibrium(
    ['Na+', 'HCO3-', 'Ca++', 'Cl-'],
    fixed_elements=['Cl-'],
)
pyequion.save_res_to_file(sys_eq, './eq_nahco3_caco3.py', '
    res')
```

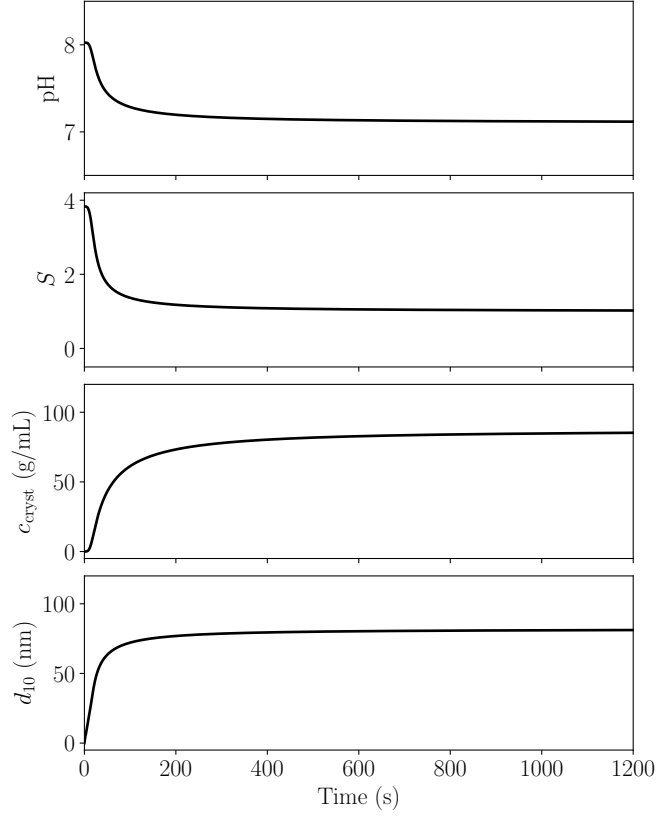Listing 6: Exporting a python function for the nonlinear residual calculation.

Figure 6: Dynamic profiles of pH, supersaturation S, crystal mass concentration c_{cryst} and particle mean size d_{10} from the simulation of $CaCO_3$ precipitation embedding equilibrium calculation from *pyequion* exported function.

5. Conclusions

Aqueous electrolyte equilibrium calculations are required in several different fields. Although there are many theoretical models available in the literature, the thermodynamic modeling of electrolyte is a controversial topic in the scientific community. In general, available software implementations of such models work well under conditions properly verified with experimental data. However, for certain operational conditions of interest, the combination of model and available parameters for calculation of reaction constants and species activity coefficients can lead to uncertainty in predictions. Many of those available softwares were written in programming languages and plat-

20

forms that may be outdated, which poses a barrier for new practitioners and researchers to extend, adapt or embed into existing mathematical models.

The package *pyequion* was developed in the Python language because of the easy of prototyping and the available plethora of scientific, data manipulation and visualization libraries, which makes research and learning workflows more simple. As a pure python package, *pyequion* can be easily installed in different computational environments. The package can be beneficial in classroom, as the students can easily visualize the equilibrium reactions and the species for a giving mixture. A web interface was developed to allow even non-programmers to run equilibrium calculations. Because the thermodynamic models are the main part in electrolyte calculations, the software was developed in a modular manner allowing researchers to implement new models and to quickly validate the predictions for a broad range of mixtures. The equilibrium calculation can be embedded in phenomenological models using the package directly, or by exporting the nonlinear residuals to a source python function. This makes the package suitable for inclusion in differential algebraic equation model. Therefore, *pyequion* makes aqueous electrolyte equilibrium more accessible to non-experts, being suitable for both academia and technological purposes. The user should be knowledgeable of limitations and constraints when using the software, as required in any aqueous electrolyte thermodynamic prediction.

Because of the relevance of aqueous electrolyte equilibrium in basic science and engineering applications, efforts on software development that facilitates this type of calculation are important. In this work, we report a pure Python package for equilibrium calculation of aqueous electrolyte systems. The package can determine the reactions and species in a given system, and it calculates equilibrium concentrations, which makes it a reliable and flexible tool for students, researchers and practitioners in technological applications.

6. Declaration of Competing Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

References

[1] C. F. Curitiba Marcellos, C. M. Senna Figueiredo, F. W. Tavares, M. Bezerra de Souza Jr, P. L. d. C. Lage, J. F. Cajaiba da Silva, A. R. Secchi, A. G. Barreto Jr, Inferring kinetic dissolution of NaCl in aqueous glycol solution using a low-cost apparatus and population balance model, The Canadian Journal of Chemical Engineering`doi: 10.1002/cjce.23774`.

[2] K. Takano, R. Gani, T. Ishikawa, P. Kolar, Computer aided design and analysis of separation processes with electrolyte systems, Comput. Chem. Eng. 24 (2-7) (2000) 645–651. `doi:10.1016/s0098-1354(00) 00383-5`.
URL `https://doi.org/10.1016/s0098-1354(00)00383-5`

[3] J. M. Prausnitz, R. N. Lichtenthaler, E. G. de Azevedo, Molecular thermodynamics of fluid-phase equilibria, Pearson Education, 1998.

[4] P. M. May, D. Rowland, Thermodynamic modeling of aqueous electrolyte systems: Current status, J. Chem. Eng. Data 62 (9) (2017) 2481–2495. `doi:10.1021/acs.jced.6b01055`.
URL `https://doi.org/10.1021/acs.jced.6b01055`

[5] P. M. May, K. Murray, JESS, A joint expert speciation systemI. Raison d'tre, Talanta 38 (12) (1991) 1409–1417. `doi:10.1016/0039-9140(91)80289-C`.
URL `http://www.sciencedirect.com/science/article/pii/003991409180289C`

[6] C. Haase, F. Dethlefsen, M. Ebert, A. Dahmke, Uncertainty in geochemical modelling of CO2 and calcite dissolution in NaCl solutions due to different modelling codes and thermodynamic databases, Appl. Geochem. 33 (2013) 306–317. `doi:10.1016/j.apgeochem.2013.03.001`.
URL `https://doi.org/10.1016/j.apgeochem.2013.03.001`

[7] D. L. Parkhurst, C. Appelo, Description of input and examples for PHREEQC version 3: A computer program for speciation, batch-reaction, one-dimensional transport, and inverse geochemical calculations, Tech. rep., US Geological Survey (2013).

[8] R. U. B. C. Consortium, et al., ScaleSoftPitzer^{TM}(Version 13.0). energy & environmental systems institute, rice university, Houston TX77005.

[9] B. Kaasa, MultiSCALE v5. 0 software, Petrotech Knowledge as, Haugesund, Norway.

[10] G. Kiranskas, J. N. Pedersen, O. Karlstrm, M. Leijnse, A. Wacker, QmeQ 1.0: An open-source Python package for calculations of transport through quantum dot devices, Computer Physics Communications 221 (2017) 317–342. doi:10.1016/j.cpc.2017.07.024.
URL http://www.sciencedirect.com/science/article/pii/S0010465517302515

[11] H. Koepke, Why python rocks for research, Hacker Monthly 8.

[12] M. Mortensen, H. P. Langtangen, High performance Python for direct numerical simulations of turbulent flows, Computer Physics Communications 203 (2016) 53–65. doi:10.1016/j.cpc.2016.02.005.
URL http://www.sciencedirect.com/science/article/pii/S0010465516300200

[13] M. R. Wright, An introduction to aqueous electrolyte solutions, John Wiley & Sons, 2007.

[14] A. B. Koudriavtsev, R. F. Jameson, W. Linert, The Law of Mass Action, Springer-Verlag Berlin Heidelberg, 2001.

[15] P. v. Debye, E. Hückel, Zur theorie der elektrolyte: I. gefrierpunktserniedrigung und verwandte erscheinungen; ii. das grenzgesetz für die elektrische leitfähigkeit (on the theory of electrolytes: I. lowering of the freezing point and related phenomena; ii. the limiting laws for the electrical conductivity), Phys. Z 24 (1923) 185.

[16] J. F. Zemaitis, D. M. Clark, M. Rafal, N. C. Scrivner, Handbook of Aqueous Electrolyte Thermodynamics, John Wiley & Sons, Inc., 1986.

doi:10.1002/9780470938416.
URL https://doi.org/10.1002/9780470938416

[17] K. S. Pitzer, G. Mayorga, Thermodynamics of electrolytes.: Ii. activity and osmotic coefficients for strong electrolytes with one or both ions univalent, in: World Scientific Series in 20th Century Chemistry, WORLD SCIENTIFIC, 1993, pp. 396–404. doi:10.1142/9789812795960_0058.
URL https://doi.org/10.1142/9789812795960_0058

[18] C. Appelo, D. Postma, Geochemistry, Groundwater and Pollution, CRC Press, 2004. doi:10.1201/9781439833544.
URL https://doi.org/10.1201/9781439833544

[19] R. Goldberg, Evaluated activity and osmotic coefficients for aqueous solutions: Thirty-six uni-bivalent electrolytes, J. Phys. Chem. Ref. Data 10 (3) (1981) 671–764. doi:10.1063/1.555646.
URL https://doi.org/10.1063/1.555646

[20] L. A. Bromley, Thermodynamic properties of strong electrolytes in aqueous solutions, AIChE J. 19 (2) (1973) 313–320. doi:10.1002/aic.690190216.
URL https://doi.org/10.1002/aic.690190216

[21] H. Meissner, J. Tester, Activity coefficients of strong electrolytes in aqueous solutions, Ind. Eng. Chem. Proc. Des. Dev. 11 (1) (1972) 128–133. doi:10.1021/i260041a025.
URL https://doi.org/10.1021/i260041a025

[22] C.-C. Chen, L. Evans, A local composition model for the excess Gibbs energy of aqueous electrolyte systems, AIChE J. 32 (3) (1986) 444–454. doi:10.1002/aic.690320311.
URL https://doi.org/10.1002/aic.690320311

[23] J.-L. Liu, C.-L. Li, A generalized debye-hückel theory of electrolyte solutions, AIP Adv. 9 (1) (2019) 015214. doi:10.1063/1.5081863.
URL https://doi.org/10.1063/1.5081863

[24] K. S. Pitzer, G. Mayorga, Thermodynamics of electrolytes. II. activity and osmotic coefficients for strong electrolytes with one or both ions univalent, J. Phys. Chem. 77 (19) (1973) 2300–2308, publisher: ACS

Publications. `doi:10.1021/j100638a009`.
URL `https://doi.org/10.1021/j100638a009`

[25] R. T. Pabalan, K. S. Pitzer, Thermodynamics of concentrated electrolyte mixtures and the prediction of mineral solubilities to high temperatures for mixtures in the system Na-k-mg-cl-SO4-OH-H2O, Geochim. Cosmochim. Acta 51 (9) (1987) 2429–2443. `doi:10.1016/0016-7037(87)90295-x`.
URL `https://doi.org/10.1016/0016-7037(87)90295-x`

[26] K. S. Pitzer, Thermodynamics of electrolytes. v. effects of higher-order electrostatic terms, J Solution Chem 4 (3) (1975) 249–265. `doi:10.1007/bf00646562`.
URL `https://doi.org/10.1007/bf00646562`

[27] L. N. Plummer, B. F. Jones, A. H. Truesdell, WATEQF-a FORTRAN IV version of WATEQ: A computer program for calculating chemical equilibrium of natural waters, Vol. 76, Department of the Interior, Geological Survey, Water Resources Division, 1976.

[28] D. L. Parkhurst, Ion-association models and mean activity coefficients of various salts, in: ACS Symposium Series, American Chemical Society, 1990, pp. 30–43. `doi:10.1021/bk-1990-0416.ch003`.
URL `https://doi.org/10.1021/bk-1990-0416.ch003`

[29] I. Grenthe, I. Puigdomenech, B. Allard, O. N. E. Agency (Eds.), Modelling in aquatic chemistry, Nuclear Energy Agency, Organisation for Economic Co-operation and Development ; Distributor, OECD Washington Center], Paris : [Washington, D.C, 1997.

[30] S. K. Lam, A. Pitrou, S. Seibert, Numba, in: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15, LLVM '15, ACM Press, New York, NY, USA, 2015. `doi:10.1145/2833157.2833162`.
URL `https://doi.org/10.1145/2833157.2833162`

[31] A. Meurer, C. P. Smith, M. Paprocki, O. ertk, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, T. Rathnayake, S. Vig, B. E. Granger, R. P. Muller, F. Bonazzi, H. Gupta, S. Vats, F. Johansson, F. Pedregosa, M. J. Curry, A. R. Terrel, S. Roučka,

A. Saboo, I. Fernando, S. Kulal, R. Cimrman, A. Scopatz, SymPy: Symbolic computing in python, PeerJ Comput. Sci. 3 (2017) e103. `doi:10.7717/peerj-cs.103`.
URL `https://doi.org/10.7717/peerj-cs.103`

[32] T. Hrbrand, T. Baumann, H. C. Moog, Validation of hydrogeochemical databases for problems in deep geothermal energy, Geotherm Energy 6 (1) (2018) 20. `doi:10.1186/s40517-018-0106-3`.
URL `https://doi.org/10.1186/s40517-018-0106-3`

[33] C. Haase, F. Dethlefsen, M. Ebert, A. Dahmke, Uncertainty in geochemical modelling of CO2 and calcite dissolution in NaCl solutions due to different modelling codes and thermodynamic databases, Appl. Geochem. 33 (2013) 306–317. `doi:10.1016/j.apgeochem.2013.03.001`.
URL `https://doi.org/10.1016/j.apgeochem.2013.03.001`

[34] C. Appelo, Principles, caveats and improvements in databases for calculating hydrogeochemical reactions in saline waters from 0 to 200 °C and 1 to 1000 atm, Appl. Geochem. 55 (2015) 62–71. `doi:10.1016/j.apgeochem.2014.11.007`.
URL `https://doi.org/10.1016/j.apgeochem.2014.11.007`

[35] E. Rodil, J. Vera, Individual activity coefficients of chloride ions in aqueous solutions of MgCl2, CaCl2 and BaCl2 at 298.2 k, Fluid Phase Equilib. 187-188 (2001) 15–27. `doi:10.1016/s0378-3812(01)00523-4`.
URL `https://doi.org/10.1016/s0378-3812(01)00523-4`

[36] A. D. Randolph, M. A. Larson, Theory of particulate processes: analysis and techniques of continuous crystallization, 2nd Edition, Academic Press, San Diego, 1988.

[37] D. Verdoes, D. Kashchiev, G. van Rosmalen, Determination of nucleation and growth rates from induction times in seeded and unseeded precipitation of calcium carbonate, J. Cryst. Growth 118 (3-4) (1992) 401–413. `doi:10.1016/0022-0248(92)90089-2`.
URL `https://doi.org/10.1016/0022-0248(92)90089-2`

[38] D. D. Nikoli, DAE tools: Equation-based object-oriented modelling, simulation and optimisation software, PeerJ Comput. Sci. 2 (2016) e54.

doi:10.7717/peerj-cs.54.
URL https://doi.org/10.7717/peerj-cs.54