# Problem Statement and Goals
# ChemCode

Samuel Crawford

January 20, 2023

Table 1: Revision History

| Date | Developer(s) | Change |
|---|---|---|
| Jan. 18, 2023 | Sam | Create document and fill in Problem, Stakeholders, and Goals sections |
| Jan. 19, 2023 | Sam | Format for Drasil upload; fill in Inputs and Outputs, Environment, and Stretch Goals sections; update Stakeholders and Goals sections; and move Environment section |
| Jan. 20, 2023 | Sam | Update Goals and Stretch Goals sections, fill in Problem Statement introduction |

## 1 Problem Statement

[You should check your problem statement with the problem statement checklist. —SS] [You can change the section headings, as long as you include the required information. —SS]

The following sections describe the problem statement of ChemCode by outlining the Problem, Inputs and Outputs, Environment, and Stakeholders.

### 1.1 Problem

Chemistry is a broad field that studies matter and its interactions [1], primarily through chemical reactions. During a chemical reaction, bonds between some substances break and new ones are formed to create new substances; these reactions are often written as chemical equations [2]. Despite new chemicals being created, all atoms from the initial substances, or "reactants", must be present in the final substances, or "products" because of the Law of Conservation of Matter [2]. This means that for a chemical equation to be useful, it must be

balanced by changing the coefficients of the substances involved in the reaction [2]. Additionally, since molecules only exist in whole numbers (since dividing a molecule changes its composition into new types of molecules), these coefficients must be whole numbers, and by convention should be as small as possible [2].

While these equations can be balanced by hand through the process of "balancing by inspection" [2], this can be time-consuming, prone to error, and inefficient, especially for more complicated chemical reactions. For each element present in the reaction, an equality can be written for the number of elements in each substance, with the reactants on one side and the products on the other, using the coefficients of each substance as the variables [3]. These equalities then form a system of linear equations that can be solved through various methods to yield a relation between each coefficient, which can then be manipulated to find the require whole numbers [2, 3]. This method can also identify reactions that are "infeasible" and balance reactions involving fractional oxidation states [3], which "are used to describe the distribution of electrons in a molecule" [4].

## 1.2 Inputs and Outputs

[Characterize the problem in terms of "high level" inputs and outputs. Use abstraction so that you can avoid details. —SS]

Input:

- A representation of a chemical equation

Output:

- A representation of the inputted chemical equation in its balanced form with the smallest whole number coefficients possible

## 1.3 Environment

[Hardware and software —SS]

ChemCode will be developed using Drasil [5], "a framework for generating high-quality documentation and code for Scientific Computing Software" [6, p. iii] by encapsulating scientific knowledge as "chunks" to be reused among projects [6]. By building this project in Drasil, relevant concepts about chemistry and systems of linear equations must first be added, along with the capability to solve these systems. Therefore, a byproduct of this project is that other programs that use chemistry and/or systems of linear equations can be made using Drasil. The implementation in Drasil places some constraints on this project.

Since Drasil is built on the idea of reusability, external libraries will be used to solve these systems of linear equations. This was previously done with ordinary differential equation (ODE) solvers, since "creating a complete ODE solver in Drasil would take considerable time, and there are already many reliable external libraries . . . tested by long use" [7, p. 24].

Additionally, Drasil can currently generate code in Python, C++, C#, Java, and Swift [7]. The scope of this project will be limited to generating code Python since it is the language in which I [First person? —SC] have the most experience. Python also supports the SciPy library [8], which is already used by Drasil to solve ODEs [7] and supports solving systems of linear equations [9].

Since both ChemCode and Drasil are purely software systems, the only hardware involved is the user's computer used to run ChemCode.

### 1.4 Stakeholders

The main stakeholder of this project is Dr. Spencer Smith, the instructor for the CAS 741 Development of Scientific Computing Software course for which this project is being completed. Dr. Smith and Dr. Jacques Carette are in charge of the Drasil project that ChemCode seeks to extend, so the implementation and development process are of significance to them. Likewise, any future developers of Drasil, including myself [Can I use the first person? —SC], are potential stakeholders of this project, since they may use features added to Drasil, such as ideas about chemistry or systems of linear equations. Jason Balaci, a fellow CAS 741 student and Drasil contributor, is of particular mention, since there may be some overlap between our projects so we may be collaborating throughout this project. I am also a stakeholder of ChemCode as the developer.

More generally, anyone in the field of chemistry in at least a high-school level may be a stakeholder of this project, as they may use this tool in their work.

## 2 Goals

The goals of this project are to develop a program that...

- can balance chemical equations (including ones with fractional oxidation states).
- can determine if a given chemical reaction is "infeasible" (i.e., not able to be balanced).
- is generated by Drasil (along with relevant documentation).
- extends Drasil by introducing the concepts from chemistry necessary to balance equations, such as elements, compounds, and reactions.
- is written in Python (see Environment).
- uses appropriate external libraries to solve systems of linear equations (see [7, Ch. 4]).

# 3 Stretch Goals

In descending order of priority, the stretch goals of this project are to...

1. add the ability to, given the amount of one substance in a reaction (in moles), calculate the amount of every other substance in the reaction (also in moles).[1]

2. add the ability to, given the amount of each reactant (in moles), determine the limiting reactant(s) in a reaction.[1]

3. add the ability to, given the amount of each reactant (in moles), determine the theoretical yield of each product and the amount of excess reactant(s).[1]

4. generate code for ChemCode in the other languages supported by Drasil. (While using external ODE solvers in Drasil, the developers "did not find a suitable library for Swift" [7, p. 24]; a similar problem may arise when using external system of linear equations solvers, meaning that ChemCode may not be generated in all five languages supported by Drasil.)

5. do the same as 1-3. but in terms of mass.[1]

6. add the ability to classify a chemical reaction as "combination (or synthesis), decomposition, combustion, single replacement, ... double replacement" [2, p. 301] or some combination of these.[1]

7. add support for phase labels.[1]

8. add support for precipitation reactions, including solubility and identifying when reactions will not take place.[1]

[1] These examples of problems related to chemical equations were taken from [2].

# References


[1] E. Gordon and Furman University, *CHM101: Chemistry and Global Awareness*. Greenville, SC, USA: LibreTexts, Jan. 2023.

[2] L. Lund and Anoka-Ramsey Community College, *Introduction to Chemistry*. Cambridge and Coon Rapids, MN, USA: LibreTexts, Jan. 2023.

[3] I. Hamid, "Balancing Chemical Equations by Systems of Linear Equations," *Applied Mathematics*, vol. 10, pp. 521–526, July 2019.

[4] Unacademy, "Fractional Oxidation States," 2023.

[5] J. Carette, D. Szymczak, B. MacLachlan, M. Niazi, S. Crawford, D. Scime, AKM11, S. Palmer, S. Smith, O. Owojaiye, A. Elwazani, Mornix, Muhammadaliog3, D. P. R. Guttapati, N. Hu, J. Wu, L. Mawarid, J. Seger, Aida, N. G. Muralidharan, Azer-X, D. Genkin, and R. Jain, "Drasil," Feb. 2021.

[6] B. MacLachlan, *A Design Language for Scientific Computing Software in Drasil*. MASc thesis, McMaster University, Hamilton, ON, Canada, June 2020.

[7] D. Chen, *Solving Higher-Order ODEs in Drasil*. MEng thesis, McMaster University, Hamilton, ON, Canada, Sept. 2022.

[8] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[9] The SciPy community, "scipy.linalg.solve," 2023.