

ChemPy: A package useful for chemistry written in Python

Björn Dahlgren¹

¹ KTH Royal Institute of Technology

DOI: [10.21105/joss.00552](https://doi.org/10.21105/joss.00552)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Submitted: 26 January 2018

Published: 28 January 2018

Licence

Authors of JOSS papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC-BY](#)).

Summary

ChemPy is a Python library which provides functions and classes for solving chemistry related problems. It includes classes for representing substances, reactions, and systems of reactions. It also includes well established formulae from physical chemistry, as well as analytic solutions to some differential equations commonly encountered in chemical kinetics. Last, but not the least, it collects parametrizations of chemical properties of substances from the literature.

The class for substances are represented by name and optionally contain information on their composition, weight, charge etc., as well as how to pretty print them in e.g. LaTeX. Both the composition and pretty printing forms can be deduced by ChemPy's parser. Reactions are represented through their stoichiometry and thermodynamic/kinetic parameters. If the stoichiometry of a reaction is unknown, ChemPy can balance it based on the composition of the substances. The classes for representing systems of reactions provide methods to analyze e.g. if there are disjoint sets of reactions, or if all are connected in the same network. The classes also offer a series of “checks” to be performed at construction, ensuring balanced reactions with sane coefficients and consistent units.

Systems of reactions can be represented as graphs, tables, systems of ordinary differential equations (chemical kinetics) or non-linear equations (chemical equilibria). The latter two forms can be solved numerically using pyodesys (Dahlgren 2018b) and pyneqsys (Dahlgren 2018a) respectively.

Thanks to the use of SymPy (Meurer et al. 2017), a user can not only solve stoichiometry problems with a single unique solution, but also under-determined systems, where the answer then contains a free parameter. In fact, most equations and parametrizations in ChemPy support, in addition to NumPy (Walt, Colbert, and Varoquaux 2011) arrays, also symbolic input, as well as arrays with explicit units. The latter allows ChemPy to check that the correct dimensionality is used based on e.g. reaction order.

Features

- Pretty printing of chemical formulae and reaction sets.
- Interactive JavaScript enabled widgets in the Jupyter notebook (Thomas et al. 2016).
- Parsing of chemical formulae, reactions and systems thereof.
- Functions for expressing systems of reactions as ordinary differential equations.
- Functions for expressing systems of equilibria as non-linear equations (including multi-phase systems).
- Analytic solutions for a selection of kinetic problems.

References

- Dahlgren, Björn. 2018a. “Pyneqsys: Solve Symbolically Defined Systems of Non-Linear Equations Numerically.” *The Journal of Open Source Software* 3 (21). The Open Journal:531. <https://doi.org/10.21105/joss.00531>.
- . 2018b. “Pyodesys: Straightforward Numerical Integration of ODE Systems from Python.” *The Journal of Open Source Software* 3 (21). The Open Journal:490. <https://doi.org/10.21105/joss.00490>.
- Meurer, Aaron, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, et al. 2017. “SymPy: Symbolic Computing in Python.” *PeerJ Computer Science* 3 (January). PeerJ:e103. <https://doi.org/10.7717/peerj-cs.103>.
- Thomas, Kluyver, Ragan-Kelley Benjamin, Pérez Fernando, Granger Brian, Bussonnier Matthias, Frederic Jonathan, Kelley Kyle, Hamrick Jessica, Grout Jason, and et al. Corlay Sylvain. 2016. “Jupyter Notebooks—a Publishing Format for Reproducible Computational Workflows.” IOS Press, 87–90. <https://doi.org/10.3233/978-1-61499-649-1-87>.
- Walt, Stéfan van der, S Chris Colbert, and Gaël Varoquaux. 2011. “The NumPy Array: A Structure for Efficient Numerical Computation.” *Computing in Science & Engineering* 13 (2). Institute of Electrical; Electronics Engineers (IEEE):22–30. <https://doi.org/10.1109/mcse.2011.37>.