The Journal of Open Source Software

ratesb_python: A Python Package for Analyzing Rate Laws in Biological Models

Longxuan Fan [1], **Joseph L. Hellerstein**[2], **and Herbert M. Sauro**[3]

1 Viterbi School of Engineering, University of Southern California, 3650 McClintock Avenue, Los Angeles, CA 90089, United States of America **2** eScience Institute, University of Washington, 3910 15th Ave NE, Seattle, WA 98195, United States of America **3** Department of Bioengineering, University of Washington, 3720 15th Ave NE, Seattle, WA 98195, United States of America

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

`ratesb_python` is a Python package that analyzes mechanistic models of biological systems that consist of networks of chemical reactions like $2H_2 + O_2 \rightarrow 2H_2O$ [with rate laws such as $k[h_2]^2[O_2]$ that describe the rate at which the reaction proceeds]. The package focuses on rate laws of reactions, algebraic expressions that specify the rate at which reactants (e.g., H_2, O_2) are converted into products (e.g., O_2). `ratesb_python` analyzes rate laws to detect errors and warnings that affect the robustness and accuracy of models that use the SBML (Systems Biology Markup Language) community standard for model model descriptions (Hucka et al., 2003).

Statement of Need

Mechanistic models in systems biology are essential tools for simulating and understanding the intricacies of complex biological systems, and a wide variety of rate laws are used. One commonly used rate law is *mass action* in which the reaction rate is proportional to the product of the concentrations of the reactants. To illustrate, consider a reaction in which m molecules of A combine with n molecules of B to produce r molecules of C, or $mA + nB \rightarrow rC$. The mass action rate law is $k * [A]^m * [B]^n$, where $[x]$ is the concentration of x and k is a constant.

The `ratesb_python` package evaluates rate laws against a library of predefined types to identify anomalies that may compromise the accuracy of mechanistic models. This process involves categorizing rate laws based on their mathematical characteristics and examining their performance within the context of the model. Such analysis enables `ratesb_python` to identify potential errors and warnings, including discrepancies in reactant usage or abnormal reaction fluxes. For example, if the rate law provided for the reaction $2H_2 + O_2 \rightarrow 2H_2O$ is $k_1[H_2]^2[O_2][H_2O]^2$ (where k_1 is a constant), `ratesb_python` reports an error since there is no defined classification for the rate law since the products are included ($[H_2O]$).

Moreover, `ratesb_python` extends beyond symbolic comparisons employed by existing tools (such as SBMLKinetics (Xu, 2023)) by implementing a **randomized polynomial identity test** for identifying and classifying rate laws. This approach provides polynomial-time complexity compared to the generally non-polynomial complexity of purely symbolic methods. The added benefit is that the algorithm is not restricted to a set of pre-coded equations, thereby enabling more flexible customization: users can readily define their own rate laws and instruct `ratesb_python` to check whether a provided rate law matches a generalized or custom-defined functional form. This functionality is especially beneficial for researchers who need to handle specialized or novel rate laws that do not necessarily fit the standard templates.

The Journal of Open Source Software

Software Description

ratesb_python analyzes rate laws to detect errors and warns about violations of best practices. Input to ratesb_python can be a file path to a model in the SBML or Antimony (Smith et al., 2009) formats, or a string in the Antimony format. The output is text and/or Python objects. Control over inputs and outputs is managed by analyzer.py.

Central to ratesb_python is the ability to classify rate laws according to widely used types such as: mass action, Michaelis-Menten, and zeroth order kinetics. ratesb_python relies heavily on approaches employed in SBMLKinetics (Xu, 2023), which uses the sympy package to do symbolic analysis of rate laws. However, ratesb_python refines and extends these approaches by using a randomized polynomial identity test, providing a more efficient and customizable framework for classifying rate laws. This functionality is complemented by custom_classifier.py, which offers users the flexibility to define and classify rate laws via a structured JSON format. This adaptability is crucial for tailoring the tool to specific research requirements, highlighting ratesb_python's commitment to user-defined customization. Default classifications are detailed in default_classifier.json.

Error and warning messages generated during the analysis are systematically managed within messages.json, ensuring users are well-informed of any issues detected during the examination process. The results of these analyses are succinctly presented through the Results class in results.py, providing users with a clear description of the findings.

Error Code Rationale and Organization

ratesb_python employs *grouped error codes* to systematically guide users in identifying and addressing issues with rate law classification. This ensures that common issues (e.g., missing reactants, invalid product usage, or typographical mistakes) are clearly distinguished from more severe problems (e.g., completely undefined rate laws). The codes also serve developers by providing a standardized mechanism for adding new checks or extending existing ones. Detailed explanations of these codes, along with examples, can be found both in the messages.json file and in the official documentation. By grouping related issues under specific ranges of codes, ratesb_python makes it easier for users to trace potential errors to their underlying causes and for developers to introduce new error or warning categories without breaking the existing structure.

Below is a summary of the error and warning codes along with their brief descriptions:

Code	Type	Brief Description
1-2	Errors	Issues with missing rate laws or expected reactants.
1001	Warning	Numeric-only rate law.
1002	Warning	Rate law unrecognized.
1003-1004	Warning	Flux relationship issues with reactants and products.
1005	Warning	Missing boundary species reactant.
1006	Warning	Non-constant parameters in rate law.
1010	Warning	Products in irreversible reaction rate law.
1020-1022	Warning	Naming conventions for parameters not followed.
1030-1037	Warning	Issues with ordering and formatting conventions in rate laws.
1040-1044	Warning	Annotations not following recommended SBO terms.

Error and warning messages to aid in rate law analysis.

Integration with Other Tools and API Capabilities

ratesb_python is designed as a flexible, modular API and standalone tool, enabling integration with various systems biology tools to facilitate rate law analysis in biological modeling projects.

Its development in Python ensures compatibility with prevalent scientific computing tools, allowing it to be added to existing systems or tailored for specific applications. It works well with tools that are widely used in the SBML community, such as `libantimony` and `libsbml`. Additionally, `ratesb_python` serves an educational purpose, offering a practical tool for computational biology courses where students can learn about rate law analysis by interacting with and modifying the API.

Future Work

Future developments for `ratesb_python` include enriching the library of checks and optimizing the performance of classification algorithms. The goal is to expand the tool's capabilities, making it a more comprehensive resource for developers and researchers alike, and to introduce customization options for error and warning management, further enhancing its utility in systems biology.

Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., Goryanin, I. I., Hedley, W. J., Hodgman, T. C., Hofmeyr, J.-H., … SBML Forum:, the rest of the. (2003). The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, *19*(4), 524–531. https://doi.org/10.1093/bioinformatics/btg015

Smith, L. P., Bergmann, F. T., Chandran, D., & Sauro, H. M. (2009). Antimony: A modular model definition language. *Bioinformatics*, *25*(18), 2452–2454. https://doi.org/10.1093/bioinformatics/btp401

Xu, J. (2023). SBMLKinetics: A tool for annotation-independent classification of reaction kinetics for SBML models. *BMC Bioinformatics*, *24*(1), 248. https://doi.org/10.1186/s12859-023-05380-3