The Journal of Open Source Software

yaml2sbml: Human-readable and -writable specification of ODE models and their conversion to SBML

Jakob Vanhoefer[1], **Marta R. A. Matos**[2], **Dilan Pathirana**[1], **Yannik Schälte**[3, 4], **and Jan Hasenauer**[1, 3, 4]

1 Faculty of Mathematics and Natural Sciences, University of Bonn, Bonn, Germany **2** The Novo Nordisk Foundation Center for Biosustainability, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark **3** Institute of Computational Biology, Helmholtz Zentrum München, Neuherberg, Germany **4** Department of Mathematics, Technical University Munich, Garching, Germany

DOI: 10.21105/joss.03215

Software

- Review
- Repository
- Archive

Editor: Kelly Rowland

Reviewers:

- @SirSharpest
- @marouenbg

Submitted: 09 April 2021
Published: 27 May 2021

Summary

Ordinary differential equations (ODE) models are used throughout natural sciences to describe dynamic processes. In systems biology, ODEs are mostly stored and exchanged using the Systems Biology Markup Language (SBML), a widely adopted community standard based on XML. The Parameter Estimation table (PEtab) format extends SBML to parameter estimation problems. A large number of software tools support simulation of SBML models and parameter estimation for PEtab problems. Specifying ODE models in SBML and parameter estimation problems in PEtab provides access to these tools. However, SBML is considered to be neither human-readable nor human-writable. An easy-to-use approach to construct the SBML/PEtab models tailored to ODE models will facilitate model generation.

In this contribution, we present `yaml2sbml`, a Python tool for converting ODE models specified in an easy-to-read and -write YAML file into SBML/PEtab. `yaml2sbml` comes with a format validator for the input YAML, a command-line interface (CLI) and a model editor to: 1) create an ODE model programmatically in Python that can then be saved as SBML, PEtab or YAML and 2) edit an ODE model previously encoded in YAML. Several examples illustrate the use of `yaml2sbml` on realistic problems.

Statement of need

ODE models have applications in many scientific fields, including biology (Hodgkin & Huxley, 1952; Lotka, 1920), epidemiology (Kermack et al., 1927), physics (Lorenz, 1963; Van der Pol & Van Der Mark, 1927), economics (Shone, 2002) or more recent neural networks (Chen et al., 2018).

SBML is a widely adopted community standard for specifying biological reaction networks (Hucka et al., 2003). sbml.org lists more than 100 software tools that accept SBML as their input format for dynamic model simulation, among them COPASI (Hoops et al., 2006), d2d (Raue et al., 2015), AMICI (Fröhlich et al., 2021) and dMod (Kaschek et al., 2019).

Model parameters can be estimated from data by formulating a likelihood function. Therefore, the system states must be mapped to measured quantities by observable functions, and a measurement noise model must be specified. The PEtab format was recently introduced to complement SBML by tab-separated value files specifying observables, measurements, experimental conditions, and estimated parameters (Schmiester et al., 2021). Currently 9 software

toolboxes support PEtab as an input format, among them COPASI, d2d, dMod and AMICI/pyPESTO. The PEtab documentation gives a complete and up-to-date list of tools.

Thanks to the aforementioned tools, model simulation or parameter estimation has become a matter of a few lines of code or clicks. However, ODE model definition is often a bottleneck, since constructing an SBML model from scratch is often tedious. Therefore, various approaches to facilitate model construction from text-based input formats or in code have been presented, as `libsbml` (Bornstein et al., 2008), `SimpleSBML` (Cannistra et al., 2015), `MOCCASIN` (Gómez et al., 2016), `Antimony` (Smith et al., 2009) and `ScrumPy` (Poolman, 2006). `MOCCASIN` translates MATLAB code into SBML. Other tools have a text-based input format that is centered around chemical reactions and not around ODEs directly (e.g. `ScrumPy`), or only offer a text-based (`Antimony`) or only a Python-based way of defining SBML models (`libsbml`, `SimpleSBML`), but not both at the same time interchangeably. Neither of these tools offer PEtab support.

Here, we present a human-readable and -writeable format tailored to ODE models that is based on YAML and can be validated and translated to SBML and PEtab via the Python tool `yaml2sbml` and a CLI. Furthermore, `yaml2sbml` comes with a format validator and a Python-based model editor that allows to generate, import, extend and export a YAML model within code.

Tool and Format

Figure 1 gives an overview of the typical workflow for model generation and conversion using `yaml2sbml`.

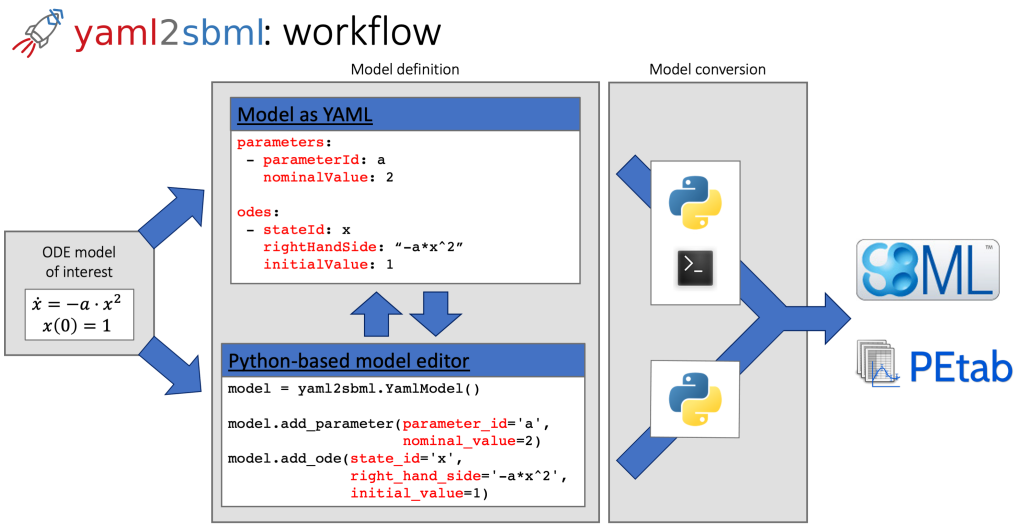

Figure 1: Typical workflow for model generation and conversion using `yaml2sbml`. The ODE is written as YAML file using any text editor, the API, or the object-oriented model editor. Both can be used interchangeably. The conversion from YAML to SBML or PEtab can be performed in Python or by the CLI.

YAML Format

Building the input format on YAML allows to parse and validate the model easily, while keeping the simplicity of a text-based format (see Figure 1). The format is organized in the blocks for different model components.

- `odes` define states, right-hand sides and initial values (as numeric values or parameters).
- `parameters` define parameters and their values. Further optional keys, e.g. optimization bounds, are written to the PEtab parameter table.
- `time` specifies the name of the time variable.
- `assignments` dynamically assign a value to a variable. These are encoded as parameter assignment rules in the SBML file.
- `functions` define functions that can be called in other model parts.
- `observables` map ODE states to measurements. Observables can be encoded as parameter assignments in the SBML file or in the PEtab observable table.
- `conditions` define different experimental setups, e.g. specific inputs. Conditions are only encoded in the PEtab condition table and do not affect the SBML file.

For more details, we refer to the format specification.

Python Tool and Command-Line Interface

The Python tool `yaml2sbml` allows one to validate models specified in the YAML input format and convert them to SBML or PEtab via

```
import yaml2sbml

# format validation
yaml2sbml.validate_yaml(yaml_file)
# SBML conversion
yaml2sbml.yaml2sbml(yaml_input_file, sbml_output_file)
# PEtab conversion
yaml2sbml.yaml2petab(yaml_input_file, PEtab_dir, model_name)
```

Validation is also performed internally before model conversion. `libsbml` (Bornstein et al., 2008) generates and validates the resulting SBML. The validator in the PEtab library checks the resulting TSV-files during conversion to PEtab.

Alongside its Python API, `yaml2sbml` comes with a CLI offering the same functionality via the commands `yaml2sbml`, `yaml2petab`, and `yaml2sbml_validate`.

`yaml2sbmls` model editor allows one to generate ODE models and programmatically add, delete, or modify model components. Further, the model editor allows one to import models from YAML and export them to YAML, SBML or PEtab.

Availability and Code Development

`yaml2sbml` is developed under the MIT license on GitHub. The tool is available on `PyPI` via `pip install yaml2sbml`. The documentation is hosted on readthedocs. Several Jupyter Notebooks contain examples covering all aspects of the tool. Code testing and continuous integration is performed via GitHub Actions.

Examples

Three notebooks use the Lotka-Volterra equations (Lotka, 1920) to introduce the different aspects of `yaml2sbml`: The Python toolbox, the CLI and the model editor. Another notebook showcases features of the input format as time-dependent or discontinuous right-hand sides.

The introductory examples are complemented by two more comprehensive examples of ODE models, which do not fit in the classical reaction network formulation for which SBML is intended.

The first application example considers the Chemical Master Equation (CME) (Gillespie, 1992), a stochastic model of (bio-)chemical processes. The Finite State Projection (FSP) truncates the infinite state space of the CME, yielding a finite-dimensional ODE (Munsky & Khammash, 2006). The example implements the FSP for a two-stage model of gene expression (Shahrezaei & Swain, 2008). `yaml2sbml` allows one to implement the 1000-dimensional ODE in less than 20 lines of code, by exploiting the rich problem structure.

The second application example considers a well-established ODE model of human glucose-insulin metabolism with 22 state variables (Sorensen, 1985). The Jupyter Notebook presents an implementation of the Sorensen model in the YAML format and uses the model editor to extend the preexisting YAML model to encode a patient-specific treatment.

Acknowledgement

The authors thank Elba Raimúndez for her help designing the logo.

Funding

J.V. was funded by the Federal Ministry of Economic Affairs and Energy (Grant no. 16KN074236) and the European Union's Horizon 2020 research and innovation program (Grant No. 686282)

M.R.A.M. was funded by the Novo Nordisk Foundation (NNF10CC1016517 and NNF14OC0009473).

Y.S. was supported by the German Research Foundation (HA7376/1-1), and the German Federal Ministry of Education and Research (FitMultiCell; 031L0159A).

D.P. was funded by the Federal Ministry of Economic Affairs and Energy (Grant no. 16KN074236)

J.H. was funded by the Federal Ministry of Education and Research (Grant no. 01ZX1705) and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2151 - 390873048.

References

Bornstein, B. J., Keating, S. M., Jouraku, A., & Hucka, M. (2008). LibSBML: An API library for SBML. *Bioinformatics*, *24*(6), 880–881. https://doi.org/10.1093/bioinformatics/btn051

Cannistra, C., Medley, K., & Sauro, H. M. (2015). SimpleSBML: A python package for creating and editing SBML models. *bioRxiv*. https://doi.org/10.1101/030312

Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural ordinary differential equations. *arXiv Preprint arXiv:1806.07366*.

Fröhlich, F., Weindl, D., Schälte, Y., Pathirana, D., Paszkowski, Ł., Lines, G. T., Stapor, P., & Hasenauer, J. (2021). AMICI: high-performance sensitivity analysis for large ordinary differential equation models. *Bioinformatics*. https://doi.org/10.1093/bioinformatics/btab227

Gillespie, D. T. (1992). A rigorous derivation of the chemical master equation. *Physica A*, *188*(1), 404–425. https://doi.org/10.1016/0378-4371(92)90283-V

Gómez, H. F., Hucka, M., Keating, S. M., Nudelman, G., Iber, D., & Sealfon, S. C. (2016). MOCCASIN: Converting MATLAB ODE models to SBML. *Bioinformatics*, *32*(12), 1905–1906. https://doi.org/10.1093/bioinformatics/btw056

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.*, *117*(4), 500–544. https://doi.org/10.1113/jphysiol.1952.sp004764

Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P., & Kummer, U. (2006). COPASI – a COmplex PAthway SImulator. *Bioinformatics*, *22*(24), 3067–3074. https://doi.org/10.1093/bioinformatics/btl485

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., … Wang, J. (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

Kaschek, D., Mader, W., Fehling-Kaschek, M., Rosenblatt, M., & Timmer, J. (2019). Dynamic modeling, parameter estimation, and uncertainty analysis in R. *J. Stat. Softw.*, *88*(10). https://doi.org/10.18637/jss.v088.i10

Kermack, W. O., McKendrick, A. G., & Walker, G. T. (1927). A Contribution to the Mathematical Theory of Epidemics. *P Roy Soc A-Math Phy*, *115*(772), 700–721. https://doi.org/10.1098/rspa.1927.0118

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of Atmospheric Sciences*, *20*(2), 130–141. https://doi.org/10.1175/1520-0469(1963)020%3C0130:DNF%3E2.0.CO;2

Lotka, A. J. (1920). Analytical note on certain rhythmic relations in organic systems. *Proceedings of the National Academy of Sciences*, *6*(7), 410–415. https://doi.org/10.1073/pnas.6.7.410

Munsky, B., & Khammash, M. (2006). The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.*, *124*(4), 044104. https://doi.org/10.1063/1.2145882

Poolman, M. G. (2006). ScrumPy: Metabolic modelling with python. *IEE Proceedings-Systems Biology*, *153*(5), 375–378. https://doi.org/10.1049/ip-syb:20060010

Raue, A., Steiert, B., Schelker, M., Kreutz, C., Maiwald, T., Hass, H., Vanlier, J., Tönsing, C., Adlung, L., Engesser, R., Mader, W., Heinemann, T., Hasenauer, J., Schilling, M., Höfer, T., Klipp, E., Theis, F. J., Klingmüller, U., Schöberl, B., & J.Timmer. (2015). Data2Dynamics: A modeling environment tailored to parameter estimation in dynamical systems. *Bioinformatics*, *31*(21), 3558–3560. https://doi.org/10.1093/bioinformatics/btv405

Schmiester, L., Schälte, Y., Bergmann, F. T., Camba, T., Dudkin, E., Egert, J., Fröhlich, F., Fuhrmann, L., Hauber, A. L., Kemmer, S., Lakrisenko, P., Loos, C., Merkt, S., Müller, W., Pathirana, D., Raimúndez, E., Refisch, L., Rosenblatt, M., Stapor, P. L., … Weindl, D. (2021). PEtab—interoperable specification of parameter estimation problems in systems biology. *PLOS Computational Biology*, *17*(1), 1–10. https://doi.org/10.1371/journal.pcbi.1008646

Shahrezaei, V., & Swain, P. S. (2008). The stochastic nature of biochemical networks. *Curr. Opin. Biotechnol.*, *19*(4), 369–374. https://doi.org/10.1016/j.copbio.2008.06.011

Shone, R. (2002). *Economic dynamics: Phase diagrams and their economic application*. Cambridge University Press.

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

Sorensen, J. T. (1985). *A physiologic model of glucose metabolism in man and its use to design and assess improved insulin therapies for diabetes* [PhD thesis]. Massachusetts Institute of Technology.

Van der Pol, B., & Van Der Mark, J. (1927). Frequency demultiplication. *Nature*, *120*(3019), 363–364. https://doi.org/10.1038/120363a0