BASICO: A simplified Python interface to COPASI

Frank T. Bergmann ORCID [1]

1 BioQUANT/COS, Heidelberg University, Heidelberg, Germany

Summary

Biological systems are highly complex and dynamic, whose behavior in response to changed conditions or perturbations cannot be predicted easily. Thus computational modeling can serve as a valuable analysis tool which allows an understanding of these systems that cannot be achieved with wet lab experiments alone. It can be used to gain a deeper understanding of the system or to answer more applied questions such as predicting drug targets. COPASI is a powerful environment for creating, editing, simulating and analyzing computational models. It provides a user-friendly GUI that is easy to use even by people who are not experts in the field. At the same time, its broad range of tasks makes it attractive for more advanced computational biologists. By using a scripting language in conjunction with COPASI, scientists can automate tasks such as parameter estimation, sensitivity analysis, optimization, and model fitting. They can also create custom scripts to perform complex operations or analyses that are not available in COPASI's GUI, such as model identification. Scripting languages can also facilitate the communication. documentation and reproducibility of computational models, by allowing scientists to share their code and results with others.

Here we introduce BASICO, a simplified Python interface to COPASI. It provides a set of functions that allow to create, edit, simulate and analyze models in a more automated way. BASICO can be easily installed using: `pip install copasi-basico`.

Statement of need

COmplex PAthway SImulator [COPASI; Hoops et al. ([2006](#)); Frank T. Bergmann et al. ([2017](#))] is a widely used program for analyzing and generating systems biology models. Its graphic user interface (GUI) makes it a quick and easy to learn tool. Since its operation requires little knowledge about mathematical concepts, it is very attractive for biologists. It incorporates very powerful methods ranging from basic time series for both deterministic and stochastic simulations, parameter estimation with a broad range of implemented local and global optimization algorithms, to more complex tasks such as time scale separation analysis, and Lyapunov exponent calculations. Import and export of models encoded in the Systems Biology Markup Language (SBML) allows a quick exchange of models with other modeling environments ([Hucka et al., 2003](#); [Keating et al., 2020](#)).

To further the powerful application range of COPASI, files prepared in the GUI can be scheduled in cluster environments. This is extremely helpful for running time consuming tasks such as optimization runs or parameter estimations. Generating these files for different parameterizations or versions of models and synchronizing is a very tedious task. That is where it is extremely helpful to use Python scripting to handle several model versions with different parameterizations, as it is often required in systems biology. Here, task parameters can be pre-defined, along with which task to run. The COPASI project uses SWIG ([Beazley, 1996](#)) to automatically generate language bindings for a variety of programming languages. These bindings however, require deep knowledge of COPASI's architecture, which makes them hard to use.

BASICO is a pure Python package, building on top of these automatically generated language bindings, in order to simplify using all of the features of COPASI. Thanks to being hosted on GitHub with its automated processes, modifications and improvements of the software can be easily integrated, and published to the community.

Being an easily installable module, BASICO can readily be integrated with other packages or pipelines to create new functionality. For example, it would be difficult to use COPASI directly for approximate Bayesian computation (ABC), but through using BASICO in the pyABC package (Schälte et al., 2022) it can be achieved.

As a scripting module BASICO lends itself for constructing large networks, as is for example done in the reproducibility study in Mendes (2023).

Documentation for BASICO along with many examples, in the form of Jupyter Notebooks can be found at https://basico.readthedocs.io/. BASICO can be readily used in the cloud using Google Colab, or mybinder.org.

Features & Functionality

BASICO provides a set of functions that allow to create, edit, simulate and analyze biochemical reaction networks. It is easy installed using:

```
pip install copasi-basico
```

From there, models can be created from scratch, or loaded from COPASI, SBML, the Simulation Experiment Description Markup Language (SED-ML) or COMBINE Archive files (Frank T. Bergmann et al., 2014). Support for SED-ML and COMBINE Archive files is provided through libSEDML (F. Bergmann, Smith, et al., 2022) and libCOMBINE (F. Bergmann, König, et al., 2022) that are used by the SWIG generated COPASI bindings.

We also provide functions, to directly access and search models from the BioModels Database (Juty et al., 2015) or JWS Online (Olivier & Snoep, 2004).

```
load_model(filename)    # loads CPS, SBML, SED-ML or COMBINE Archive files
load_biomodel(model_id) # loads models from the BioModels Database
```

Of course the wrappers for the REST API to JWS Online or the BioModels Database can also be readily used by other Python packages to obtain the SBML models. This is done for example by SBMLtoODEjax (Etcheverry et al., 2023)

Once a model is loaded all of COPASI's analysis methods can be used. Running simulations are a core feature of COPASI, so we started BASICO with implementing time course simulations and steady state analysis.

```
load_example('brusselator')       # load an example model
df = run_time_course(duration=10) # simulate the model for 10 time units
```

Here, run_time_course returns a pandas.DataFrame (pandas development team, 2023) with the results of the simulation. Feature requests from the community led us to add further analysis methods, and so we added parameter estimation, optimizations, sensitivity analysis and parameter scans.

The most recent additions include the automation of profile likelihood calculations, that for a given parameter estimation result, automatically generates profile likelihood plots. This is another example, that would have been quite cumbersome to do without BASICO. Following the approach of Schaber (Schaber, 2012), BASICO generates parameter scans (running a local optimization method) for each parameter to be estimated, can run these individual models in parallel, and generate the likelihood plots from the generated data.

Target Audience

This package is intended for researchers in the field of systems biology, who are interested in either creating and understanding new computational models, or to automate the analysis of existing models. The package is also very useful as teaching tool.

Acknowledgements

References

Beazley, D. M. (1996). SWIG: An easy to use tool for integrating scripting languages with c and c++. *Proceedings of the 4th Conference on USENIX Tcl/Tk Workshop, 1996 - Volume 4*, 15–15. http://dl.acm.org/citation.cfm?id=1267498.1267513

Bergmann, Frank T., Adams, R., Moodie, S., Cooper, J., Glont, M., Golebiewski, M., Hucka, M., Laibe, C., Miller, A. K., Nickerson, D. P., & others. (2014). COMBINE archive and OMEX format: One file to share all information to reproduce a modeling project. *BMC Bioinformatics*, *15*(1), 1–9. https://doi.org/10.1186/s12859-014-0369-z

Bergmann, Frank T., Hoops, S., Klahn, B., Kummer, U., Mendes, P., Pahle, J., & Sahle, S. (2017). COPASI and its applications in biotechnology. *Journal of Biotechnology*, *261*, 215–220. https://doi.org/10.1016/j.jbiotec.2017.06.1200

Bergmann, F., König, M., Karr, J., Keegan, L., & Plesníková, J. (2022). *Sbmlteam/libCombine: Release 0.2.20* (Version v0.2.20). Zenodo. https://doi.org/10.5281/zenodo.7408732

Bergmann, F., Smith, L., Nickerson, D., Garny, A., Medley, K., & Marakasov, D. (2022). *Fbergmann/libSEDML: Release 2.0.32* (Version v2.0.32). Zenodo. https://doi.org/10.5281/zenodo.6619620

Etcheverry, M., Levin, M., Moulin-Frier, C., & Oudeyer, P.-Y. (2023). *SBMLtoODEjax: Efficient simulation and optimization of ODE SBML models in JAX*. https://doi.org/10.48550/arXiv.2307.08452

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.

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

Juty, N., Ali, R., Glont, M., Keating, S., Rodriguez, N., Swat, M., Wimalaratne, S., Hermjakob, H., Le Novère, N., Laibe, C., & Chelliah, V. (2015). BioModels: Content, Features, Functionality, and Use: BioModels: Content, Features, Functionality, and Use. *CPT: Pharmacometrics & Systems Pharmacology*, *4*(2), 55–68. https://doi.org/10.1002/psp4.3

Keating, S. M., Waltemath, D., König, M., Zhang, F., Dräger, A., Chaouiya, C., Bergmann, F. T., Finney, A., Gillespie, C. S., Helikar, T., Hoops, S., Malik-Sheriff, R. S., Moodie, S. L., Moraru, I. I., Myers, C. J., Naldi, A., Olivier, B. G., Sahle, S., Schaff, J. C., … Hucka, M.

(2020). SBML Level 3: An extensible format for the exchange and reuse of biological models. *Molecular Systems Biology*, *16*(8), e9110. https://doi.org/10.15252/msb.20199110

Mendes, P. (2023). Reproducibility and FAIR principles: The case of a segment polarity network model. *Frontiers in Cell and Developmental Biology*, *11*. https://doi.org/10.3389/fcell.2023.1201673

Olivier, B. G., & Snoep, J. L. (2004). Web-based kinetic modelling using JWS online. *Bioinformatics*, *20*(13), 2143–2144. https://doi.org/10.1093/bioinformatics/bth200

pandas development team. (2023). *Pandas-dev/pandas: pandas* (Version v2.0.3). Zenodo. https://doi.org/10.5281/zenodo.8092754

Schaber, J. (2012). Easy parameter identifiability analysis with COPASI. *Biosystems*, *110*(3), 183–185. https://doi.org/10.1016/j.biosystems.2012.09.003

Schälte, Y., Klinger, E., Alamoudi, E., & Hasenauer, J. (2022). pyABC: Efficient and robust easy-to-use approximate bayesian computation. *Journal of Open Source Software*, *7*(74), 4304. https://doi.org/10.21105/joss.04304