The Journal of Open Source Software

Episimlab: a Python package for modeling epidemics

Ethan Ho[1], Kelly Pierce[1], Zhanwei Du[2], Remy Pasco[2], Xutong Wang[2], and Lauren Ancel Meyers[2]

1 Texas Advanced Computing Center (TACC) - University of Texas at Austin **2** The University of Texas at Austin

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

Computational models play a crucial role in our scientific understanding of, and preparedness for, infectious disease epidemics. These *in silico* disease models simulate real-world transmission dynamics and can be applied to tasks such as early detection of novel pandemics, improving situational awareness during periods of high prevalence, and estimating efficacy of intervention strategies. These modeling approaches have proven valuable in responding to emergent epidemics such as the H1N1 flu pandemic, the Ebola epidemic, the Zika virus pandemic, and the recent COVID-19 pandemic (Rivers et al., 2019). For example, during the early waves of the COVID-19 pandemic, compartmental disease models were instrumental for projecting case counts and hospitalizations in Austin, Texas (K. Pierce et al., 2020; K. A. Pierce et al., 2020). As more data on case incidence, hospitalizations, and pathogen genomics become available, disease modelers are able to simulate increasingly complex disease dynamics. Developing such complex compartmental models is time-consuming. In addition, relatively few model implementations share a common software framework or application flow, which complicates efforts to reproduce model outputs. Therefore, there is a urgent need for more robust cyberinfrastructure in the field of epidemic modeling.

Episimlab is a software development kit (SDK) written in Python that seeks to address this problem by providing a flexible framework for developing compartmental disease models. Models in Episimlab are collections of modular components, known as "processes," which can be added, removed, or replaced to modify the dynamics of the simulated disease. Data is passed between processes, usually as N-dimensional arrays, using a standardized interface provided by the xarray-simlab (Bovy et al., 2021) package. In practice, this means that Episimlab supports development of models that:

1. Implement any compartment structure that can be represented as a graph, including cyclic graphs
2. Have many input parameters with variable dimensionality, such as recovery rates that depend on age and risk factors, or contact patterns that depend on location.
3. Simulate interventions, including dynamic interventions such as administering vaccines only when case incidence exceeds a threshold
4. Incorporate one or more data sources that are too large to load eagerly into CPU memory.

The package is designed to provide a minimal but extensible boilerplate; it includes several pre-packaged models that the user can run in a few lines of code. When the user implements different or more complex transmission dynamics, they can do so by adding or replacing processes in the model. In addition, Episimlab provides a scalable and performant runtime environment for model execution, thanks to integration with packages in the PyData stack such as Dask (Dask Development Team, 2016), Xarray (Hoyer & Hamman, 2017), and xarray-simlab (Bovy et al., 2021). Finally, Episimlab provides a standard for packaging, versioning, and sharing models, using Python's built-in class attributes.

Statement of Need

Episimlab is a Python package that provides a common framework for rapid development of complex compartmental disease models. It provides sufficient boilerplate such that the user can quickly instantiate a basic compartmental model. Basic models such as the SIR model - containing the three compartments susceptible (S), infected (I), and recovered (R) - are not unique to Episimlab; they have a long history of use ever since their origin in the early 20th century (Kermack & McKendrick, 1927; Ross, 1916; Ross & Hudson, 1917a, 1917b). More recently, various implementations of compartmental models have been made publicly available and easily usable as open-source software packages (Jacob, 2021; Miller & Ting, 2019; Simon Dobson, 2021; Van den Broeck et al., 2011). These packages are valuable because they simplify execution of many simple compartmental models, but their usage is limited to the discrete handful of model structures that are published with each package. In addition, such projects rarely support complex models containing more than 5 or 6 compartments, in part because complex models are difficult to develop and reproduce.

Inspired by previous works, Episimlab aims to support development of models with arbitrary complexity. It gives the user flexibility to define key components of their compartmental model, such as the dimensionality of the Markov state space (Grimmett & Stirzaker, 2020), the number of compartments, structure and rate of transitions between compartments, and custom stochastic behavior. Therefore, Episimlab supports not only the handful of compartmental models that are included in the package, but also an indefinite number of model structures which can be tailored for specific modeling use cases. This is accomplished by enforcing a modular paradigm of model development. The package provides a library of lightweight Python classes, known as processes in the API, which comprise a model when combined with other processes. The core process `ComptModel` is the only process shared by all Episimlab models. `ComptModel` implements a Gillespie algorithm that supports deterministic or stochastic discrete-time Markov chain models (Gagniuc, 2017; Gillespie, 1977), using a generic model of compartmental disease. Therefore, Episimlab does not support models that run in continuous time, are defined using differential equations, or are agent-based. The pre-packaged models included in Episimlab draw transition matrix values from Poisson distributions, but model developers can easily replace the Poisson with other distribution functions.

Of note, comparable software such as `epydemic` (Simon Dobson, 2021) and GLEaMviz (Van den Broeck et al., 2011) also support generic models of compartmental disease. They do not, however, support arbitrary dimensionality in input variables or in the Markov state matrix, thereby limiting their usage to simulations that run in fixed-dimensional space.

Episimlab was originally designed with epidemiological use cases in mind via collaboration with data scientists and epidemiologists in the UT Austin COVID-19 Modeling Consortium. Specifically, prototypes of Episimlab were used in studies projecting hospital burden due to the COVID-19 epidemic in Austin, Texas (K. Pierce et al., 2020; K. A. Pierce et al., 2020). Two of the pre-packaged models - `partition_v1` and `vaccine` - were developed by epidemiologists in the Consortium and migrated to Episimlab (Lachmann et al., 2021; Yang et al., 2020).

Although the package was originally intended for use by epidemiologists, it is useful for anyone developing compartmental models of disease spread. For students, it provides a minimal boilerplate for developing basic models in pure Python. It introduces and reinforces best practices in object-oriented software development such as modularity and reproducibility. For disease modeling experts, Episimlab provides a platform that supports a wide variety of modeling use cases. Simple models can be easily adapted into more complex ones, encouraging a model development approach that is rapid, iterative, and organic. Under the hood, Episimlab leverages concurrency in xarray-simlab, dataset chunking in Dask, and accelerated matrix math in xarray, so models are performant even when using large input datasets. The standardized structure of models and processes simplifies code sharing, therby promoting collaborative development within and between disease modeling teams.

Dependencies

xarray xarray-simlab dask networkx matplotlib

Related Packages

epydemic

`epydemic` is a Python package that provides a common framework for building models of epidemic processes (Simon Dobson, 2021). It supports simulations that are discrete-time synchronous or continuous-time stochastic (Gillespie). Like Episimlab, it supports a generic model for compartmental disease, allowing for flexibility in the compartmental model structure. In addition, it ships with several basic compartmental models such as SIR, SIS, and SEIR.

EoN (Epidemics on Networks)

Epidemics on Networks (EoN) is a Python package that simulates disease dynamics for SIR and SIS models (Miller & Ting, 2019). The package includes numerical solutions for 20 different differential equation models, and supports complex contagions using the Gillespie algorithm (Gillespie, 1977).

Eir

`Eir` is a Python package that simulates epidemics using compartmental models (Jacob, 2021). It includes 4 distinct models with different mobility dynamics. In additon, it provides utilities for inspecting transmission chains, analyzing state histories, and visualizing simulation results.

GLEaMviz

The Global Epidemic and Mobility (GLEaMviz) framework is a software system for simulating spread of emergent diseases (Van den Broeck et al., 2011). The framework is comprised of two major software components: the client-side graphical user interface (GUI) and the GLEaMviz simulation engine. The simulation engine incorporates high-resolution demographic and mobility data and supports a generic model of compartmental disease, while the front-end GUI provides an intuitive interface for specifying the desired model structure.

Acknowledgements

We would like to thank the UT Austin COVID-19 Modeling Consortium for their collaborative support. In particular, we acknowledge Dr. Lauren Ancel Meyers and her team for their guidance throughout the development process. We also thank Alyssa Cantu for editing the manuscript.

This work is supported by CDC Contract 75D-301-19-C-05930 and NIH Grant 3R01AI151176-01S1.

References

Bovy, B., McBain, G. D., Gailleton, B., & Lange, R. (2021). *Benbovy/xarray-simlab: 0.5.0* (Version 0.5.0) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.4469813

Dask Development Team. (2016). *Dask: Library for dynamic task scheduling*. https://dask.org

Gagniuc, P. A. (2017). *Markov chains: From theory to implementation and experimentation*. Wiley. ISBN: 9781119387558

Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, *81*(25), 2340–2361. https://doi.org/10.1021/j100540a008

Grimmett, G., & Stirzaker, D. (2020). *Probability and random processes*. OUP Oxford. ISBN: 9780192586865

Hoyer, S., & Hamman, J. (2017). Xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*, *5*(1). https://doi.org/10.5334/jors.148

Jacob, M. (2021). Eir: A python package for epidemic simulation. *Journal of Open Source Software*, *6*(62), 3247. https://doi.org/10.21105/joss.03247

Kermack, W. O., & McKendrick, A. G. (1927). A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *115*(772), 700–721. https://doi.org/10.1098/rspa.1927.0118

Lachmann, M., Bouchnita, A., Woody, S., Pasco, R., Johnson-Leon, Maureen, Johnson, K., Fox, S. J., & Meyers, L. A. (2021). *COVID-19 scenario projections for austin, texas - july 19, 2021*. https://doi.org/10.15781/gc9z-9h56

Miller, J. C., & Ting, T. (2019). EoN (epidemics on networks): A fast, flexible python package for simulation, analytic approximation, and analysis of epidemics on networks. *Journal of Open Source Software*, *4*(44), 1731. https://doi.org/10.21105/joss.01731

Pierce, K. A., Ho, E., Wang, X., Pasco, R., Du, Z., Zynda, G., Song, J., Wells, G., Fox, S. J., & Meyers, L. A. (2020). Early COVID-19 pandemic modeling: Three compartmental model case studies from texas, USA. *Computing in Science & Engineering*, *23*(1), 25–34. https://doi.org/10.1109/MCSE.2020.3037033

Pierce, K., Ho, E., Wang, X., Pasco, R., Du, Z., Fox, S., Zynda, G., Song, J., & Meyers, L. A. (2020). COVID-19 healthcare demand projections: Beaumont-port arthur MSA texas. *UT COVID-19 Model. Consortium*. https://doi.org/10.13140/RG.2.2.14066.53443

Rivers, C., Chretien, J.-P., Riley, S., Pavlin, J. A., Woodward, A., Brett-Major, D., Berry, I. M., Morton, L., Jarman, R. G., Biggerstaff, M., & others. (2019). Using "outbreak science" to strengthen the use of models during epidemics. *Nature Communications*, *10*(1), 1–3. https://doi.org/10.1038/s41467-019-11067-2

Ross, R. (1916). An application of the theory of probabilities to the study of a priori pathometry.—part i. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *92*(638), 204–230. https://doi.org/10.1098/rspa.1916.0007

Ross, R., & Hudson, H. P. (1917a). An application of the theory of probabilities to the study of a priori pathometry.—part II. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *93*(650), 212–225. https://doi.org/10.1098/rspa.1917.0014

Ross, R., & Hudson, H. P. (1917b). An application of the theory of probabilities to the study of a priori pathometry.—part III. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *93*(650), 225–240. https://doi.org/10.1098/rspa.1917.0015

Simon Dobson. (2021). *Epydemic* (Version 1.7.2) [Computer software]. https://pyepydemic.readthedocs.io/

Van den Broeck, W., Gioannini, C., Gonçalves, B., Quaggiotto, M., Colizza, V., & Vespignani, A. (2011). The GLEaMviz computational tool, a publicly available software to explore

realistic epidemic spreading scenarios at the global scale. *BMC Infectious Diseases*, *11*(1), 1–14. https://doi.org/10.1186/1471-2334-11-37

Yang, H., Sürer, Ö., Duque, D., Morton, D. P., Singh, B., Fox, S. J., Pasco, R., Pierce, K., Rathouz, P., Du, Z., Pignone, M., Escott, M. E., Adler, S. I., Johnston, S. C., & Meyers, L. A. (2020). Design of COVID-19 staged alert systems to ensure healthcare capacity with minimal closures. *medRxiv*. https://doi.org/10.1101/2020.11.26.20152520