The Journal of Open Source Software

epipack: An infectious disease modeling package for Python

Benjamin F. Maier[1, 2]

1 Institute for Theoretical Biology, Humboldt-University of Berlin, Philippstr. 13, D-10115 Berlin **2** Robert Koch Institute, Nordufer 20, D-13353 Berlin

DOI: 10.21105/joss.03097

Software

- Review
- Repository
- Archive

Editor: Frederick Boehm

Reviewers:

- @chrisvoncsefalvay
- @esteinig

Submitted: 04 March 2021
Published: 01 April 2021

Summary

Analyzing the spread of infectious diseases by means of compartmental mathematical models has been an active area of research for almost a century (Kermack & McKendrick, 1991), (Keeling & Rohani, 2011), (Anderson & May, 2010). Since the emergence of the coronavirus disease 2019 pandemic in early 2020, the field has seen yet another considerable boost in interest. Researchers have since been working on a large number of different models that are used to forecast case numbers, to analyze the implications of different contact structures between individual people, or to discuss the influence of various mitigation and containment strategies, to only name a few applications (Estrada, 2020).

The complexity of detailed epidemiological models often mitigates simple replication and/or adaptation to local circumstances. Typically, modifications are quickly thought up, but their influence on system properties such as the epidemic threshold or outbreak size are less clear. Variations in model formulation often entail the reimplementation of simulation and analysis frameworks for every model iteration, taking up valuable time and resources for debugging and reanalysis. Furthermore, researchers often need to cross-check their results by implementing both deterministic well-mixed models as well as models that consider explicit contact structures (*i.e.*, static or temporal networks). Last but not least, analytical derivations are often done using separate computer algebra systems.

epipack solves the raised issues by offering a simple, process-based framework that allows researchers to quickly prototype and modify compartmental epidemiological models and to investigate their behavior based on analytical, numerical, stochastic, and network-based simulations, facilitated by a visualization framework and parsimonious, customizable interactives.

Here, the overarching design principle focuses on defining epidemiological models via reaction processes or reaction events, from which ordinary differential equations (ODEs) and simulation classes are generated automatically. This allows the user to transfer implemented models quickly between analytical, numerical, or stochastical formulations.

epipack provides four base classes to accomodate building models for different analysis methods:

- *EpiModel*: Define a model based on transition, birth, death, fission, fusion, or transmission reactions to integrate the ordinary differential equations (ODEs) of the corresponding well-mixed system numerically or simulate the system using Gillespie's algorithm (Gillespie, 1977). Process rates can be numerical functions of time and system state.
- *SymbolicEpiModel*: Define a model based on transition, birth, death, fission, fusion, or transmission reactions. Obtain the ODEs, fixed points, Jacobian, and the Jacobian's eigenvalues at fixed points as symbolic expressions using *sympy* (Meurer et al.,

2017). Process rates can be symbolic expressions of time and system state. Set numerical parameter values and integrate the ODEs numerically, or simulate the stochastic formulation using Gillespie's algorithm (Gillespie, 1977).

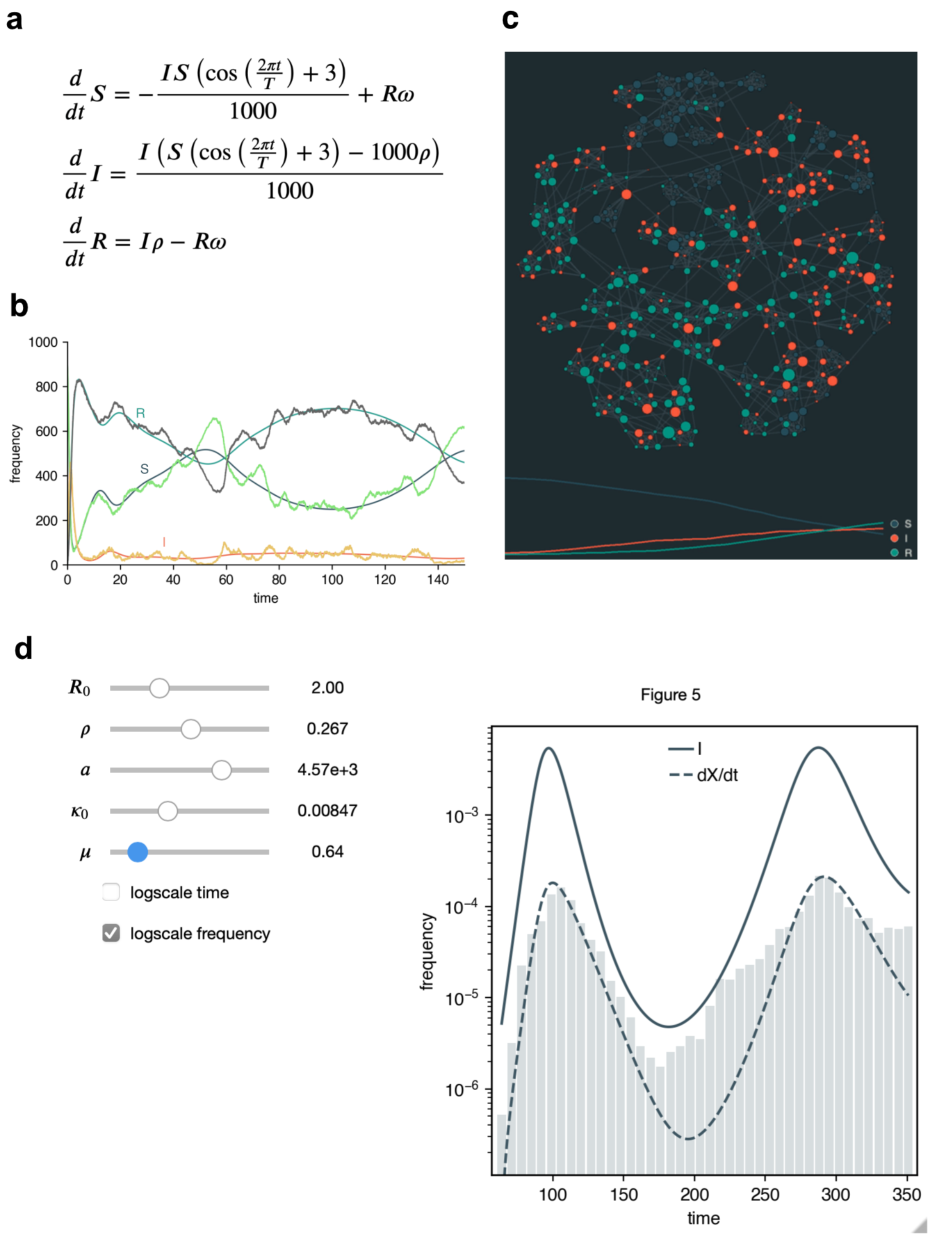
- *StochasticEpiModel*: Define a network model based on node transition and link transmission reactions. Add conditional link transmission reactions. Simulate your model on any (un)directed, (un)weighted static/temporal network, or in a well-mixed system. We make use of a generalized version of the tree-based rejection sampling algorithm recently proposed (St-Onge et al., 2019) and the accompanying implementation of *SamplableSet* (St-Onge, 2019). This algorithm is a variant of Gillespie's continuous-time algorithm, which *epipack* focuses on because discrete-time approximative simulation methods like the individual-based update algorithm are known to behave problematically at times (Givan et al., 2011), (Maier, 2020), (Kiss et al., 2017). The class further allows to define chained (*i.e.*, conditional) reactions using which public health interventions such as contact tracing can be simulated. The *StochasticEpiModel* class is comparable to the `Gillespie_simple_contagion` function of the *EoN* (Epidemics on Networks) package (Miller & Ting, 2019), which does not yet, however, support temporal networks or conditional reactions at the time of writing.
- *MatrixEpiModel*: This class is a static-rate version of the *EpiModel* class that runs faster on more complex models (e.g. for meta-population reaction-diffusion systems) by making use of scipy's implementation of sparse matrices (Virtanen et al., 2020).

Moreover, we provide a simple OpenGL-based visualization framework to animate stochastic simulations on networks, lattices, well-mixed systems, or reaction-diffusion systems. The research process is further facilitated by interactive analysis widgets for *Jupyter* notebooks that give immediate visual feedback regarding a system's inner workings.

epipack and its usage is exhaustively documented, with the documentation being available at epipack.benmaier.org and in the repository.

While other reaction-based modeling packages exist, most focus either purely on ODE systems (e.g. *ChemPy* (Dahlgren, 2018)) or simulations and analyses on static network systems (for instance *EoN* (Epidemics on Networks) (Miller & Ting, 2019), *epydemic* (Dobson, 2017), and *GraphTool* (Peixoto, 2014)). One exception is the *EpiModel* package, which is, however, only available for the R language (Jenness et al., 2018).

To the best of our knowledge, *epipack* is the first open source software suite for Python that offers extensive model building and analysis frameworks for both mean-field and network models with a simple and intuitive API. It thus presents a valuable tool for researchers of the infectious disease modeling community.



$$\frac{d}{dt}S = -\frac{IS\left(\cos\left(\frac{2\pi t}{T}\right)+3\right)}{1000} + R\omega$$

$$\frac{d}{dt}I = \frac{I\left(S\left(\cos\left(\frac{2\pi t}{T}\right)+3\right)-1000\rho\right)}{1000}$$

$$\frac{d}{dt}R = I\rho - R\omega$$

R_0 2.00
ρ 0.267
a 4.57e+3
κ_0 0.00847
μ 0.64
☐ logscale time
☑ logscale frequency

Figure 1: Example use cases of *epipack*. (a) Equations that have been generated automatically in a *Jupyter* notebook from a *SymbolicEpiModel* instance that was built via reaction processes (here, a temporally forced SIRS model in a population of 1000 individuals). (b) Stochastic simulation and result from the ODE integration of the model defined for panel a. Both stochastic and deterministic results have been obtained from the same model instance. (c) A screen shot from a stochastic simulation visualization of a model on a static network. (d) Screen shot of the interactive *Jupyter* notebook widget for a custom-built *SymbolicEpiModel*, plotted against data.

Acknowledgments

BFM is financially supported as an *Add-On Fellow for Interdisciplinary Life Science* by the Joachim Herz Stiftung. BFM wants to thank A. Burdinski and K. Ledebur for valuable feedback as well as J. S. Juul for the productive collaboration on a code base that heavily inspired this project.

References

Anderson, R. M., & May, R. M. (2010). *Infectious diseases of humans: Dynamics and control* (Reprinted). Oxford Univ. Press. ISBN: 978-0-19-854040-3

Dahlgren, B. (2018). ChemPy: A package useful for chemistry written in Python. *Journal of Open Source Software*, *3*(24), 565. https://doi.org/10.21105/joss.00565

Dobson, S. (2017). *Epydemic: Epidemic simulations on networks in Python*. https://pyepydemic.readthedocs.io/en/latest/index.html

Estrada, E. (2020). COVID-19 and SARS-CoV-2. Modeling the present, looking at the future. *Physics Reports*, *869*, 1–51. https://doi.org/10.1016/j.physrep.2020.07.005

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

Givan, O., Schwartz, N., Cygelberg, A., & Stone, L. (2011). Predicting epidemic thresholds on complex networks: Limitations of mean-field approaches. *Journal of Theoretical Biology*, *288*, 21–28. https://doi.org/10.1016/j.jtbi.2011.07.015

Jenness, S. M., Goodreau, S. M., & Morris, M. (2018). EpiModel: An R Package for Mathematical Modeling of Infectious Disease over Networks. *Journal of Statistical Software*, *84*. https://doi.org/10.18637/jss.v084.i08

Keeling, M. J., & Rohani, P. (2011). *Modeling infectious diseases in humans and animals*. Princeton University Press. ISBN: 978-1-4008-4103-5

Kermack, W. O., & McKendrick, A. G. (1991). Contributions to the mathematical theory of epidemics I. *Bulletin of Mathematical Biology*, *53*(1-2), 33–55. https://doi.org/10.1007/BF02464423

Kiss, I. Z., Miller, J. C., & Simon, P. L. (2017). *Mathematics of Epidemics on Networks: From Exact to Approximate Models* (1st ed. 2017). Springer International Publishing. https://doi.org/10.1007/978-3-319-50806-1

Maier, B. F. (2020). *Spreading Processes in Human Systems*. https://doi.org/10.18452/20950

Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., … Scopatz, A. (2017). SymPy: Symbolic computing in Python. *PeerJ Computer Science*, *3*, e103. https://doi.org/10.7717/peerj-cs.103

Miller, J., & 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

Peixoto, T. P. (2014). The graph-tool python library. *Figshare*. https://doi.org/10.6084/m9.figshare.1164194

St-Onge, G. (2019). *SamplableSet*. https://github.com/gstonge/SamplableSet

St-Onge, G., Young, J.-G., Hébert-Dufresne, L., & Dubé, L. J. (2019). Efficient sampling of spreading processes on complex networks using a composition and rejection algorithm. *Computer Physics Communications*, *240*, 30–37. https://doi.org/10.1016/j.cpc.2019.02.008

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S. J. van der, Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2