




pypfilt: a particle filter for Python

Robert Moss ¹

¹ Melbourne School of Population and Global Health, The University of Melbourne, Australia

DOI: [10.21105/joss.06276](https://doi.org/10.21105/joss.06276)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Patrick Diehl](#)  

Reviewers:

- [@tbrown122387](#)
- [@Karangupta1994](#)

Submitted: 16 January 2024

Published: 03 April 2024

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

Summary

Mathematical models are used to simulate real-world systems in many scientific fields. These models can be fitted to real-time data, and used to generate *probabilistic forecasts* that describe how the system will behave in the future and convey the *uncertainty* in these predictions. Particle filters are a class of Sequential Monte Carlo (SMC) methods ([Doucet et al., 2001](#)) that have been used in many scientific fields for real-time forecasting, with the COVID-19 pandemic being one of the most recent and high-profile examples. These methods can be used to estimate model parameters and state as new data become available (“online estimation”).

`pypfilt` is a Python package for online estimation and forecasting that implements several particle filters. It was developed to enable real-time seasonal influenza forecasting in Australia, for which we won two national innovation awards ([Pyne, 2018](#)), and played a key role in generating forecasts that have supported Australia’s COVID-19 response ([Moss et al., 2023](#)). The package is deliberately generic and readily applicable to other domains.

Statement of need

Particle filters are provided by Python packages that implement a range of inference methods, such as PyMC ([Abril-Pla et al., 2023](#)), pyro ([Bingham et al., 2019](#)), and stonesoup ([Hiscocks et al., 2023](#)); accompany textbooks, such as particles ([Chopin & Papaspiliopoulos, 2020](#)) and filterpy ([Labbe, 2022](#)); and focus on other applications, such as SMCPy ([Leser & Wang, 2023](#)). However, `pypfilt` appears to be unique in supporting all of the following: (a) online state estimation for time-series forecasting; (b) arbitrary state-space models; (c) non-analytic likelihood functions; (d) control over memory usage for large-scale applications; (e) a declarative approach for defining and running forecasts; and (f) reproducible results. In brief:

- *Forecasts are generated efficiently.* Repeated calculations are avoided when new data are received (online estimation), even when previous data are updated or corrected, due to a sophisticated caching system.
- *Complex models are easily defined.* Models only need to define the state vector, with arbitrarily nested fields and dimensions, and define an update rule for the state vector.
- *Likelihood functions are unconstrained.* They do not need to be differentiable, and can inspect both the current state and any previous states.
- *Memory usage is flexible.* The particle filter can retain only a sliding window of previous states, and only record states at coarse intervals, so that forecasts with large numbers of particles and/or very small time-steps do not exceed available memory.
- *Forecasts are simple to define and run.* Forecasts are defined in plain-text configuration files, enforcing a clear separation between model implementation and experiment (e.g., choice of prior distributions, input data, particle filter settings).

- *Reproducibility is ensured.* Output files include all data and metadata required to reproduce the original results.

Additional features include supporting scalar and calendar time scales, providing a range of resampling strategies (Kitagawa, 1996) and summary statistics, post-regularisation (Musso et al., 2001), and measuring forecast performance with Continuous Ranked Probability Scores (CRPS) (Hyndman & Athanasopoulos, 2021). It also supports *scenario modelling* — simulating from multiple prior distributions and comparing matching particles in each ensemble — which provides additional decision-support capabilities beyond those provided by real-time forecasts (Moss et al., 2020; Shearer et al., 2020).

A suite of almost 150 *test cases* (comprising more than 6,000 lines of code) runs automatically every time the code is updated. This includes tests which verify that outputs are reproducible, and tests which verify that outputs are identical whether or not the particle filter resumed from a previously-cached state.

Availability and usage

This is available as a Python package on [PyPI](https://pypi.org/) and can be installed using `pip`. It deliberately supports older versions of Python (≥ 3.8), NumPy (≥ 1.17), and SciPy (≥ 1.4). The documentation is available at <https://pypfilt.readthedocs.io/>, and includes a [Getting Started tutorial](#) that demonstrates how to construct a model, fit it to data, and generate forecasts (see [Figure 1](#)). In this tutorial we use the Lorenz-63 system of ordinary differential equations (which has chaotic solutions) to show how post-regularisation can greatly improve forecast performance (see [Figure 2](#)), and to highlight how easy it is to define and run a suite of forecasts.

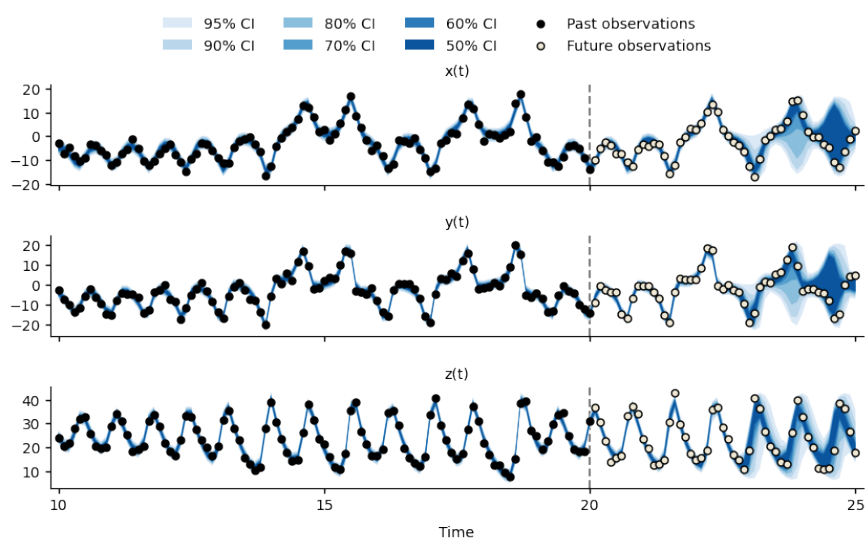


Figure 1: An example forecast at time $t = 20$ for the Lorenz-63 system. This figure was generated using the `pypfilt.plot` module.

Ongoing research projects

Beginning in 2015, `pypfilt` was developed to support real-time seasonal influenza forecasting in Australia (Moss et al., 2016, 2017, 2018; Moss, Zarebski, Carlson, et al., 2019; Moss, Zarebski, Dawson, et al., 2019; Zarebski et al., 2017), and has been used to support the Australian Government’s response to COVID-19 (Moss et al., 2023; Price et al., 2020).

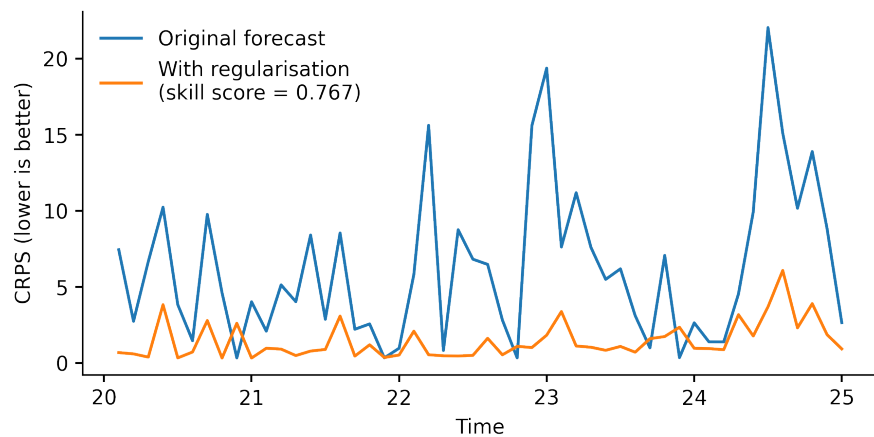


Figure 2: An example of using Continuous Ranked Probability Scores (CRPS) to compare forecast performance. Post-regularisation improves performance by 76.7% in this example.

Acknowledgements

Peter Dawson, James M. McCaw, David J. Price, and Alexander E. Zarebski contributed helpful comments and suggestions. Package development was supported by the Defence Science and Technology Group project “Bioterrorism Preparedness Strategic Research Initiative 07/301”, and by Australian National Health and Medical Research Council (NHMRC) Centres for Research Excellence (PRISM, 1078068; APPRISE, 1116530; SPECTRUM, 1170960). RM was supported by an NHMRC APPRISE Research Fellowship (1116530).

References

- Abril-Pla, O., Andreani, V., Carroll, C., Dong, L., Fonnesebeck, C. J., Kochurov, M., Kumar, R., Lao, J., Luhmann, C. C., Martin, O. A., Osthege, M., Vieira, R., Wiecki, T., & Zinkov, R. (2023). PyMC: A modern, and comprehensive probabilistic programming framework in Python. *PeerJ Computer Science*, 9, e1516. <https://doi.org/10.7717/peerj-cs.1516>
- Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P. A., Horsfall, P., & Goodman, N. D. (2019). Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*, 20, 28:1–28:6. <http://jmlr.org/papers/v20/18-403.html>
- Chopin, N., & Papaspiliopoulos, O. (2020). *An introduction to Sequential Monte Carlo*. Springer. <https://doi.org/10.1007/978-3-030-47845-2>
- Doucet, A., Freitas, N. de, & Gordon, N. (Eds.). (2001). *Sequential Monte Carlo methods in practice* (1st ed.). Springer. <https://doi.org/10.1007/978-1-4757-3437-9>
- Hiscocks, S., Harrald, O., Barr, J., Perree, N., Vladimirov, L., Green, R., Rosoman, O., etfrogers-dstl, idorrington-dstl, Glover, T., Hunter, E., Wright, J., gawebb-dstl, Harris, M., Fraser, B., spike, Pritchett, H., jjosborne-dstl, Carniglia, P., ... Hiles, J. (2023). *Dstl/stone-soup: v1.1 release* (Version v1.1). Zenodo. <https://doi.org/10.5281/zenodo.8308177>
- Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice* (3rd ed.). OTexts. <https://otexts.com/fpp3>
- Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1), 1–25. <https://doi.org/10.1080/10618600.1996.10474692>

- Labbe, R. R. (2022). FilterPy - Kalman filters and other optimal and non-optimal estimation filters in Python. In *GitHub repository*. GitHub. <https://github.com/rlabbe/filterpy>
- Leser, P., & Wang, M. (2023). SMCPy - Sequential Monte Carlo with Python. In *GitHub repository*. GitHub. <https://github.com/nasa/SMCPy>
- Moss, R., Fielding, J. E., Franklin, L. J., Stephens, N., McVernon, J., Dawson, P., & McCaw, J. M. (2018). Epidemic forecasts as a tool for public health: Interpretation and (re)calibration. *Australian and New Zealand Journal of Public Health*, 42(1), 69–76. <https://doi.org/10.1111/1753-6405.12750>
- Moss, R., Price, D. J., Golding, N., Dawson, P., McVernon, J., Hyndman, R. J., Shearer, F. M., & McCaw, J. M. (2023). Forecasting COVID-19 activity in Australia to support pandemic response: May to October 2020. *Scientific Reports*, 13(1). <https://doi.org/10.1038/s41598-023-35668-6>
- Moss, R., Wood, J., Brown, D., Shearer, F. M., Black, A. J., Glass, K., Cheng, A. C., McCaw, J. M., & McVernon, J. (2020). Coronavirus disease model to inform transmission-reducing measures and health system preparedness, Australia. *Emerging Infectious Diseases*, 26(12), 2844–2853. <https://doi.org/10.3201/eid2612.202530>
- Moss, R., Zarebski, A. E., Carlson, S. J., & McCaw, J. M. (2019). Accounting for healthcare-seeking behaviours and testing practices in real-time influenza forecasts. *Tropical Medicine and Infectious Disease*, 4(1), 12. <https://doi.org/10.3390/tropicalmed4010012>
- Moss, R., Zarebski, A. E., Dawson, P., Franklin, L. J., Birrell, F. A., & McCaw, J. M. (2019). Anatomy of a seasonal influenza epidemic forecast. *Communicable Diseases Intelligence*, 43, 1–14. <https://doi.org/10.33321/cdi.2019.43.7>
- Moss, R., Zarebski, A., Dawson, P., & McCaw, J. M. (2016). Forecasting influenza outbreak dynamics in Melbourne from Internet search query surveillance data. *Influenza and Other Respiratory Viruses*, 10(4), 314–323. <https://doi.org/10.1111/irv.12376>
- Moss, R., Zarebski, A., Dawson, P., & McCaw, J. M. (2017). Retrospective forecasting of the 2010–14 Melbourne influenza seasons using multiple surveillance systems. *Epidemiology and Infection*, 145(1), 156–169. <https://doi.org/10.1017/S0950268816002053>
- Musso, C., Oudjane, N., & Le Gland, F. (2001). Improving regularised particle filters. In *Sequential Monte Carlo methods in practice* (pp. 247–271). Springer. https://doi.org/10.1007/978-1-4757-3437-9_12
- Price, D. J., Shearer, F. M., Meehan, M. T., McBryde, E. S., Moss, R., Golding, N., Conway, E. J., Dawson, P., Cromer, D., Wood, J., Abbott, S., McVernon, J., & McCaw, J. M. (2020). Early analysis of the Australian COVID-19 epidemic. *eLife*, 9, e58785. <https://doi.org/10.7554/eLife.58785>
- Pyne, C. (2018). Disease forecasting system takes out National Innovation Awards. In *Media release*. Australian Department of Defence. <https://www.minister.defence.gov.au/media-releases/2018-05-11/disease-forecasting-system-takes-out-national-innovation-awards>
- Shearer, F. M., Moss, R., McVernon, J., Ross, J. V., & McCaw, J. M. (2020). Infectious disease pandemic planning and response: Incorporating decision analysis. *PLOS Medicine*, 17, e1003018. <https://doi.org/10.1371/journal.pmed.1003018>
- Zarebski, A. E., Dawson, P., McCaw, J. M., & Moss, R. (2017). Model selection for seasonal influenza forecasting. *Infectious Disease Modelling*, 2(1), 56–70. <https://doi.org/10.1016/j.idm.2016.12.004>