The Journal of Open Source Software

aimz: Scalable probabilistic impact modeling

Eunseop Kim [1]

1 Eli Lilly and Company, United States

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 05 October 2025
Published: unpublished

Summary

aimz is a Python library for scalable probabilistic impact modeling, enabling assessment of intervention effects on outcomes while providing an intuitive interface for fitting Bayesian models, drawing posterior samples, generating large-scale posterior predictive simulations, and estimating interventional effects with minimal boilerplate. It combines the usability of general machine learning APIs with the flexibility of probabilistic programming through a single high-level object (`ImpactModel`). Built atop JAX (Bradbury et al., 2018) and NumPyro (Phan et al., 2019), it supports (minibatch) stochastic variational inference (SVI) and Markov chain Monte Carlo sampling, just-in-time (JIT)-compiled parallel predictive streaming to chunked Zarr (Miles et al., 2020) stores exposed through Xarray (Hoyer & Hamman, 2017), and first-class intervention handling for effect estimation. Integrated MLflow (Zaharia et al., 2018) support enables experiment tracking and model lineage. These design choices reduce bespoke glue code and enable reproducible, high-throughput analyses on large datasets, while supporting rapid iteration and experimentation.

Statement of need

Applied analytics workflows often require: (1) fitting probabilistic models to large scale datasets, (2) generating posterior and posterior predictive samples for calibrated uncertainty, and (3) estimating intervention effects under explicit structural modifications. While core probabilistic programming frameworks (e.g., NumPyro, PyMC (Oriol et al., 2023), Stan (Carpenter et al., 2017)) offer mature inference algorithms, recurring engineering tasks—such as streaming large predictive draws to disk, structuring outputs, coordinating intervention scenarios, managing device resources, and logging experiments—are typically reimplemented in an ad hoc manner. General machine learning libraries (e.g., scikit-learn (Pedregosa et al., 2011)) lack native Bayesian sampling or causal intervention semantics, while many causal inference toolkits emphasize causal graph discovery or fixed-form treatment effect routines rather than scalable sampling workflows.

`aimz` consolidates these infrastructural concerns within a single object (`ImpactModel`) that provides: probabilistic model tracing and argument binding; SVI or MCMC with automatic posterior sample management; JIT-compiled, sharded posterior predictive sampling with concurrent streaming to Zarr stores surfaced as Xarray objects; structured intervention ("do-operation") application; and optional MLflow integration for experiment tracking. The familiar "`fit` / `predict` / `sample`" interface further eases integration with machine learning tooling, emerging Model Context Protocol pipelines, and AI agents that work with simple estimator-like semantics. This unification reduces redundant glue code and minimizes potential failure points in applications such as marketing mix modeling, policy evaluation, and attribution of program impacts.

Existing impact or uplift modeling libraries (e.g., domain-specific frameworks such as Meridian (Google Meridian Marketing Mix Modeling Team, 2025), Robyn (Zhou et al., 2024), or PyMC-Marketing (PyMC Labs, 2025)) typically standardize on a constrained family of time-series or marketing response models and a fixed inference stack, making it difficult to deviate from their built-in assumptions without forking code or re-implementing infrastructure. `aimz` instead pursues generality—accepting arbitrary NumPyro model functions and multiple inference strategies—while still avoiding boilerplate through streamed predictive simulation, structured outputs, and intervention orchestration. By elevating scalable posterior predictive simulation and intervention effect estimation to first-class capabilities, `aimz` lowers the barrier between exploratory probabilistic modeling and production-grade Bayesian impact analysis.

References

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/jax-ml/jax

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., & Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, *76*, 1–32.

Google Meridian Marketing Mix Modeling Team. (2025). *Meridian: Marketing mix modeling* (Version 1.2.1). https://github.com/google/meridian

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

Miles, A., Kirkham, J., Durant, M., Bourbeau, J., Onalan, T., Hamman, J., Patel, Z., shikharsg, Rocklin, M., dussin, raphael, Schut, V., Andrade, E. S. de, Abernathey, R., Noyes, C., sbalmer, bot, pyup.io, Tran, T., Saalfeld, S., Swaney, J., … Banihirwe, A. (2020). *Zarr-developers/zarr-python: v2.4.0* (Version v2.4.0). Zenodo. https://doi.org/10.5281/zenodo.3773450

Oriol, A.-P., Virgile, A., Colin, C., Larry, D., J., F. C., Maxim, K., Ravin, K., Jupeng, L., C., L. C., A., M. O., Michael, O., Ricardo, V., Thomas, W., & Robert, Z. (2023). PyMC: A modern and comprehensive probabilistic programming framework in python. *PeerJ Computer Science*, *9*, e1516. https://doi.org/10.7717/peerj-cs.1516

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

Phan, D., Pradhan, N., & Jankowiak, M. (2019). Composable effects for flexible and accelerated probabilistic programming in NumPyro. *arXiv Preprint arXiv:1912.11554*.

PyMC Labs. (2025). *Marketing statistical models in PyMC* (Version 0.16.0). https://github.com/pymc-labs/pymc-marketing

Zaharia, M. A., Chen, A., Davidson, A., Ghodsi, A., Hong, S. A., Konwinski, A., Murching, S., Nykodym, T., Ogilvie, P., Parkhe, M., Xie, F., & Zumar, C. (2018). Accelerating the Machine Learning Lifecycle with MLflow. *IEEE Data Eng. Bull.*, *41*, 39–45. https://api.semanticscholar.org/CorpusID:83459546

Zhou, G., Lares, B., Skokan, I., & Sentana, L. (2024). *Robyn: Semi-automated marketing mix modeling (MMM) from meta marketing science* (Version 3.12.0). https://github.com/facebookexperimental/Robyn