




FlipPy: Pythonic Probabilistic Programming

Mark K. Ho ^{1*} and Carlos G. Correa ^{1*}

¹ New York University ¶ Corresponding author * These authors contributed equally.

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

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

Editor: 

Submitted: 30 August 2025

Published: unpublished

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

Probabilistic programming languages provide a user-friendly interface for specifying complex probabilistic models and inference algorithms over those models. Within psychology and cognitive science, probabilistic programming languages have been used to formally characterize human concept learning, social reasoning, intuitive physics, and planning, among many other higher-level cognitive phenomena ([Griffiths et al., 2024](#)). Meanwhile, Python is a widely used, general-purpose programming language that is commonly taught and increasingly used by students in psychology, neuroscience, and computer science. While several probabilistic programming frameworks currently exist in the scientific Python eco-system, these require beginners to learn new framework-specific syntax for specifying models and not all of them are universal (i.e., allow specification and inference over any computable distribution).

Statement of need

FlipPy is a package for specifying probabilistic programs directly in Python syntax and allows users to write any computable distribution. Existing Python-based probabilistic programming libraries tend to be optimized for specific use cases in machine learning and require users to learn specialized syntax, which can be challenging for beginners (e.g., ([Abril-Pla et al., 2023](#)), ([Bingham et al., 2019](#))). The API of FlipPy is intentionally beginner-friendly and heavily inspired by that of WebPPL ([N. D. Goodman & Stuhlmüller, 2014](#)), a Javascript-based probabilistic programming language that is widely used in computational cognitive science. Like WebPPL and the LISP-based Church ([N. Goodman et al., 2012](#)), FlipPy lets users specify probabilistic models as programs in a deterministic “host language” (Python) that is augmented with sample and observe statements. A custom interpreter treats these statements as sampling and conditioning events, which is sufficient for universality ([Van De Meent et al., 2018](#)). This interpreter is based on a continuation-passing style transform, allowing program execution to be forked when samples are taken, and resumed or halted to facilitate caching ([Ritchie et al., 2016](#)). Importantly, FlipPy itself is entirely Python-based: the codebase is implemented in Python and it performs the probabilistic execution necessary for inference in Python. This means that FlipPy can seamlessly interoperate with other Python code before, during, and after a user performs inference, including in Jupyter notebooks ([Kluyver et al., 2016](#)).

FlipPy has been designed to facilitate rapid prototyping and “hackability” while also being as accessible as possible for users who have only basic familiarity with programming in Python (e.g., behavioral scientists who are new to computational modeling). Because the specification language is Python itself, the programs are highly readable and models can be expressed using syntactic constructs like branching, function calls, for loops, etc. ([Van Rossum & others, 2007](#)). This makes the library especially valuable for teaching abstract probabilistic concepts in a concrete, iterative manner by starting with simpler models and adding complexity. Earlier versions of FlipPy have been used in undergraduate- and graduate-level courses on computational cognitive science.

Example Usage

The following is a simple program that samples (with `flip`) and observes (with `condition`) from Bernoulli distributions.

```
from flippy import infer, condition, flip
```

```
@infer
def model(p):
    x = flip(p)
    y = flip(p)
    condition(x >= y)
    return x + y
```

```
model(0.5)
```

	Element	Probability
0	2	0.333
1	1	0.333
2	0	0.333

Research projects using FlipPy

As noted, FlipPy has so far been primarily used for teaching, but the authors and their colleagues are using the library in several ongoing projects related to decision-making and social cognition. For example, Zhang et al. (2025) used FlipPy to model interactions between pragmatic reasoning and hierarchical Bayesian inference during the interpretation of generic utterances.

Acknowledgements

We acknowledge support from Thomas Griffiths during the genesis of this project.

Abril-Pla, O., Andreani, V., Carroll, C., Dong, L., Fonnesbeck, C. J., Kochurov, M., Kumar, R., Lao, J., Luhmann, C. C., Martin, O. A., & others. (2023). PyMC: A modern, and comprehensive probabilistic programming framework in python. *PeerJ Computer Science*, 9, e1516.

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Horsfall, P., & Goodman, N. D. (2019). Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*, 20(28), 1–6.

Goodman, N. D., & Stuhlmüller, A. (2014). *The Design and Implementation of Probabilistic Programming Languages*. <http://dippl.org>.

Goodman, N., Mansinghka, V., Roy, D. M., Bonawitz, K., & Tenenbaum, J. B. (2012). Church: A language for generative models. *arXiv Preprint arXiv:1206.3255*.

Griffiths, T. L., Chater, N., & Tenenbaum, J. B. (2024). *Bayesian models of cognition: Reverse engineering the mind*. MIT Press.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., & others. (2016). Jupyter notebooks—a publishing format for reproducible computational workflows. In *Positioning and power in academic publishing: Players, agents and agendas* (pp. 87–90). IOS press.

- 70 Ritchie, D., Stuhlmüller, A., & Goodman, N. (2016). C3: Lightweight Incrementalized MCMC
71 for Probabilistic Programs using Continuations and Callsite Caching. *Proceedings of the*
72 *19th International Conference on Artificial Intelligence and Statistics*, 28–37.
- 73 Van De Meent, J.-W., Paige, B., Yang, H., & Wood, F. (2018). An introduction to probabilistic
74 programming. *arXiv Preprint arXiv:1809.10756*.
- 75 Van Rossum, G., & others. (2007). Python programming language. *USENIX Annual Technical*
76 *Conference*, 41, 1–36.
- 77 Zhang, M. Y., Leslie, S.-J., Rhodes, M., & Ho, M. K. (2025). Learning about inductive
78 potential from generic statements. *Proceedings of the 47th Annual Conference of the*
79 *Cognitive Science Society*.

DRAFT