The Journal of Open Source Software

ArviZ a unified library for exploratory analysis of Bayesian models in Python

Ravin Kumar[4], **Colin Carroll**[1], **Ari Hartikainen**[2], **and Osvaldo Martin**[3]

1 Freebird Inc., United States **2** Aalto University, Department of Civil Engineering, Espoo, Finland **3** Instituto de Matemática Aplicada San Luis, UNSL-CONICET. Ejército de los Andes 950, 5700 San Luis, Argentina **4** Carbon IT LLC, United States

DOI: 10.21105/joss.01143

Software

- Review
- Repository
- Archive

Submitted: 23 December 2018
Published: 15 January 2019

Summary

While conceptually simple, Bayesian methods can be mathematically and numerically challenging. Probabilistic programming languages (PPLs) implement functions to easily build Bayesian models together with efficient automatic inference methods. This helps separate the model building from the inference, allowing practitioners to focus on their specific problems and leaving PPLs to handle the computational details for them (Bessiere, Mazer, Ahuactzin, & Mekhnacha, 2013; Daniel Roy, 2015; Ghahramani, 2015). The inference process generates a *posterior distribution* — which has a central role in Bayesian statistics — together with other distributions like the *posterior predictive distribution* and the *prior predictive distribution*. The correct visualization, analysis, and interpretation of these distributions is key to properly answer the questions that motivate the inference process.

When working with Bayesian models there are a series of related tasks that need to be addressed besides inference itself:

- Diagnoses of the quality of the inference
- Model criticism, including evaluations of both model assumptions and model predictions
- Comparison of models, including model selection or model averaging
- Preparation of the results for a particular audience

Successfully performing such tasks are central to the iterative and interactive modeling process. These tasks require both numerical and visual summaries to help statisticians or practitioners analyze visual summaries. In the words of Persi Diaconis (Diaconis, 2011) "Exploratory data analysis seeks to reveal structure, or simple descriptions in data. We look at numbers or graphs and try to find patterns. We pursue leads suggested by background information, imagination, patterns perceived, and experience with other data analyses".

For these reasons we introduce ArviZ, a Python package for exploratory analysis of Bayesian models. ArviZ aims to be a package that integrates seamlessly with established probabilistic programming languages like PyStan ("Stan," n.d.), PyMC (Salvatier, Wiecki, & Fonnesbeck, 2016), Edward (Tran et al., 2017, 2016), emcee (Foreman-Mackey, Hogg, Lang, & Goodman, 2013), Pyro (Bingham et al., 2018), and easily integrated with novel or bespoke Bayesian analyses. Where the aim of the probabilistic programming languages is to make it easy to build and solve Bayesian models, the aim of the ArviZ library is to make it easy to process and analyze the results from the Bayesian models. We

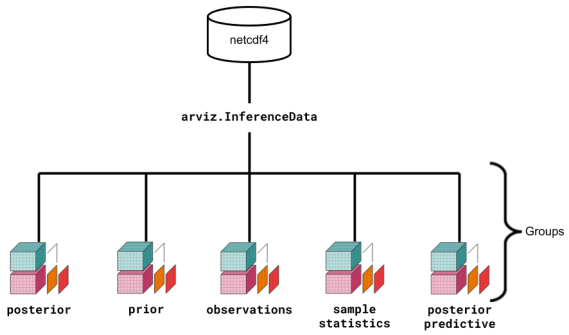

Figure 1: Relationship between netCDF, az.InferenceData, and xarray

hope ArviZ will become a key Python tool for Bayesian data analysis by allowing users to focus on problems from their domain knowledge and not on computational details.

Bayesian inference produces naturally high dimensional data. By storing each type of data resulting from PPLs as an xarray (Hoyer & Hamman, 2017) dataset, ArviZ provides labeled querying of the data, efficient algorithms, and persistent metadata. These datasets are stored together on disk and in code using netCDF4 (Brown, Folk, Goucher, Rew, & Dubois, 1993; R. Rew & Davis, 1990) groups, which are themselves built with HDF5, and allows for well supported serialization. This functionality is implemented in the InferenceData class (see Figure 1). In addition to the listed benefits of using netCDF and xarray, by using a single data structure all statistical and visualization functions need to be implemented only once.

In addition to common plots for Bayesian analysis including a trace plot and forest plot, ArviZ implements other visualizations such as a plot for posterior predictive checks, a pair plot, and a parallel coordinate plot (Gabry, Simpson, Vehtari, Betancourt, & Gelman, 2017). Additionally, it supports a number of statistical checks, such as calculating the effective sample size, the r-hat statistic, Pareto-smoothed importance sampling leave-one-out cross validation (PSIS-LOO-CV) (Vehtari, Gelman, & Gabry, 2015), and widely applicable information criterion (WAIC) (Watanabe, 2013).

Funding

Work by Osvaldo Martin was supported by CONICET-Argentina and ANPCyT-Argentina (PICT-0218).

Acknowledgments

We thank the PyMC3 Community — especially Adrian Seyboldt, Junpeng Lao, and Thomas Wiecki — as well as the Stan community — especially Allen Riddell . We also would like to extend thanks to all the ArviZ contributors, and the contributors of the libraries used to build ArviZ — particularly xarray, matplotlib, pandas, and numpy.

Example Plots

Examples of ArviZ's plotting functionality are shown in Figure 2 through Figure 5. Additional examples can be found in the ArviZ documentation.

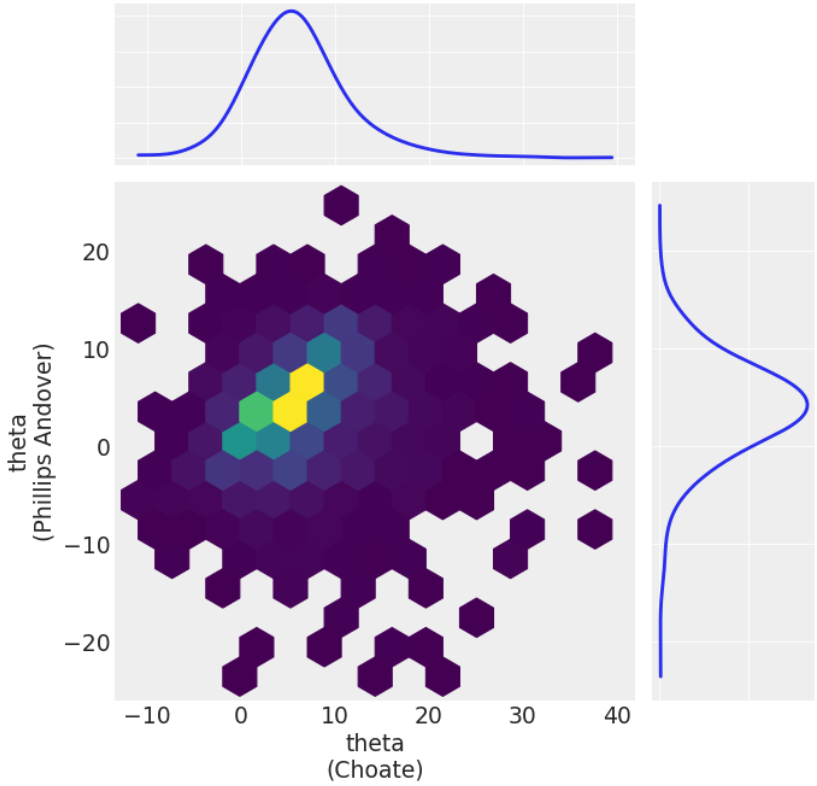

Figure 2: Bivariate Hexbin Plot with marginal distributions

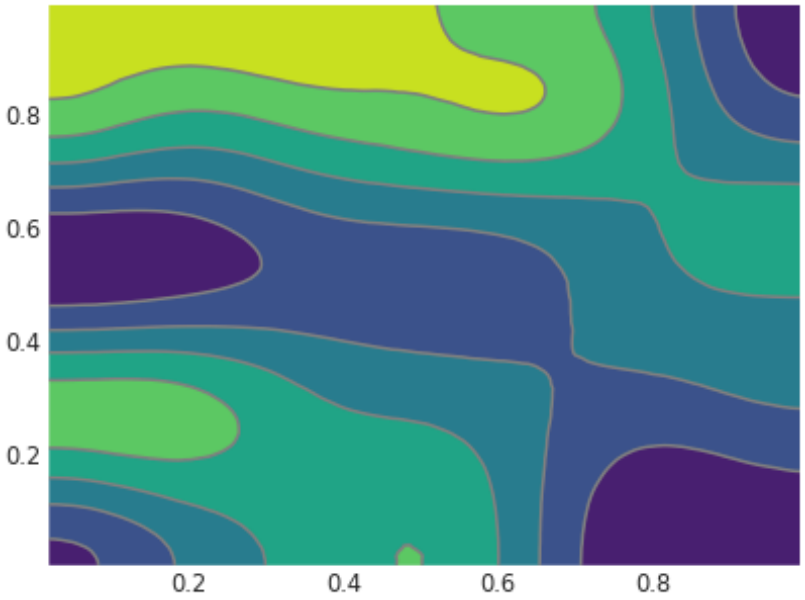

Figure 3: 2D Kernel Density Estimation

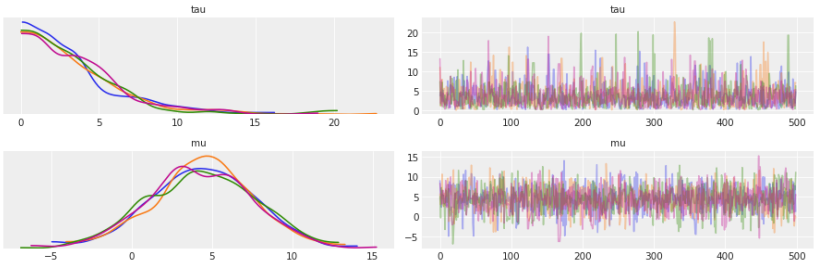

Figure 4: Markov Chain Monte Carlo Trace

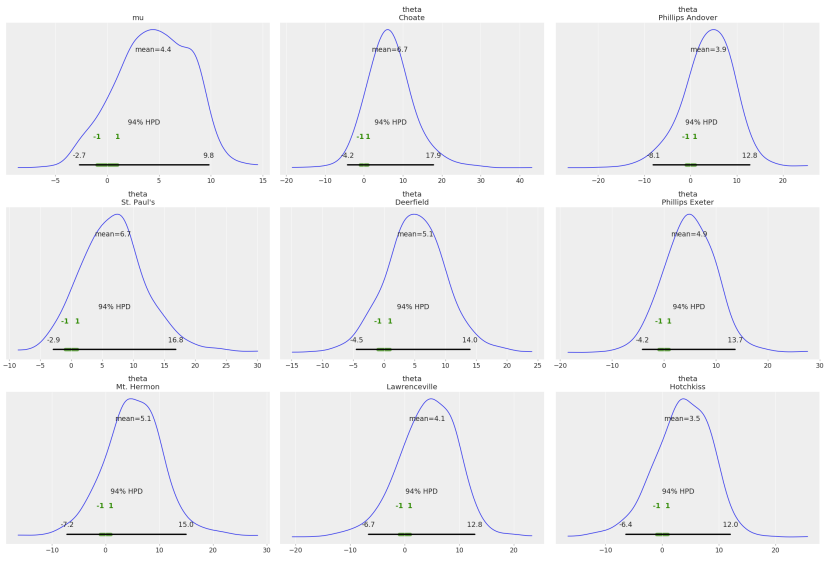

Figure 5: John Kruschke styled Posterior Distribution

References

Bessiere, P., Mazer, E., Ahuactzin, J. M., & Mekhnacha, K. (2013). *Bayesian Programming* (1 edition.). Boca Raton: Chapman and Hall/CRC.

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., et al. (2018). Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*. Retrieved from http://arxiv.org/abs/1810.09538

Brown, S. A., Folk, M., Goucher, G., Rew, R., & Dubois, P. F. (1993). Software for portable scientific data management. *Computers in Physics*, *7*(3), 304–308. doi:10.1063/1.4823180

Daniel Roy. (2015). Probabilistic Programming. *http://probabilistic-programming.org/wiki/Home*. http://probabilistic-programming.org/wiki/Home.

Diaconis, P. (2011). Theories of Data Analysis: From Magical Thinking Through Classical Statistics. In *Exploring Data Tables, Trends, and Shapes* (pp. 1–36). John Wiley & Sons, Ltd. doi:10.1002/9781118150702.ch1

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. (2013). Emcee: The MCMC Hammer. *PASP*, *125*, 306–312. doi:10.1086/670067

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., & Gelman, A. (2017). Visualization in Bayesian workflow. *arXiv:1709.01449 [stat]*. Retrieved from http://arxiv.org/abs/1709.01449

Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature*, *521*(7553), 452–459. doi:10.1038/nature14541

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

Rew, R., & Davis, G. (1990). NetCDF: An interface for scientific data access. *IEEE Computer Graphics and Applications*, *10*(4), 76–82. doi:10.1109/38.56302

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic Programming in Python Using PyMC3. *PeerJ Computer Science*, *2*, e55. doi:10.7717/peerj-cs.55

Stan: A Probabilistic Programming Language | Carpenter | Journal of Statistical Software. (n.d.). doi:10.18637/jss.v076.i01

Tran, D., Hoffman, M. D., Saurous, R. A., Brevdo, E., Murphy, K., & Blei, D. M. (2017). Deep Probabilistic Programming. *arXiv:1701.03757 [cs, stat]*. Retrieved from http://arxiv.org/abs/1701.03757

Tran, D., Kucukelbir, A., Dieng, A. B., Rudolph, M., Liang, D., & Blei, D. M. (2016). Edward: A library for probabilistic modeling, inference, and criticism. *arXiv:1610.09787 [cs, stat]*. Retrieved from http://arxiv.org/abs/1610.09787

Vehtari, A., Gelman, A., & Gabry, J. (2015). Practical Bayesian Model Evaluation Using Leave-One-out Cross-Validation and WAIC. *arXiv:1507.04544 [stat]*. Retrieved from http://arxiv.org/abs/1507.04544

Watanabe, S. (2013). A Widely Applicable Bayesian Information Criterion. *Journal of Machine Learning Research*, *14*, 867–897. Retrieved from http://arxiv.org/abs/1208.6338