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

21 November 2018

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 (Daniel Roy 2015; Bessiere et al. 2013; 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: A Probabilistic Programming Language | Carpenter | Journal of Statistical Software," n.d.), PyMC (Salvatier, Wiecki, and Fonnesbeck 2016), Edward (Tran et al. 2017; Tran et al. 2016), emcee (Foreman-Mackey et al. 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 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 and 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 (R. Rew and Davis 1990; Brown et al. 1993) 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.

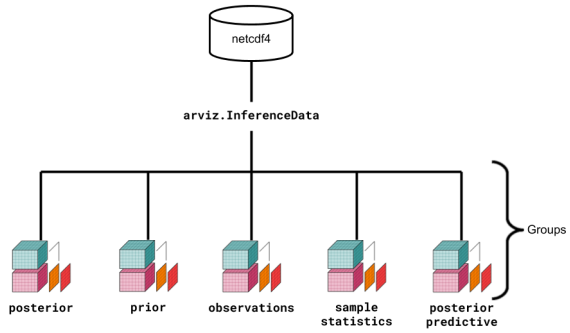

Figure 1: Relationship between netCDF, az.InferenceData, and xarray

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 et al. 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, and Gabry 2015), and widely applicable information criterion (WAIC) (Watanabe 2013).

Example plots

A portion of ArviZ's functionality is shown in Figure 2. ArviZ supports more plotting and inference diagnostics, in addition to the ones shown here. Additional examples can be found ArviZ documentation gallery

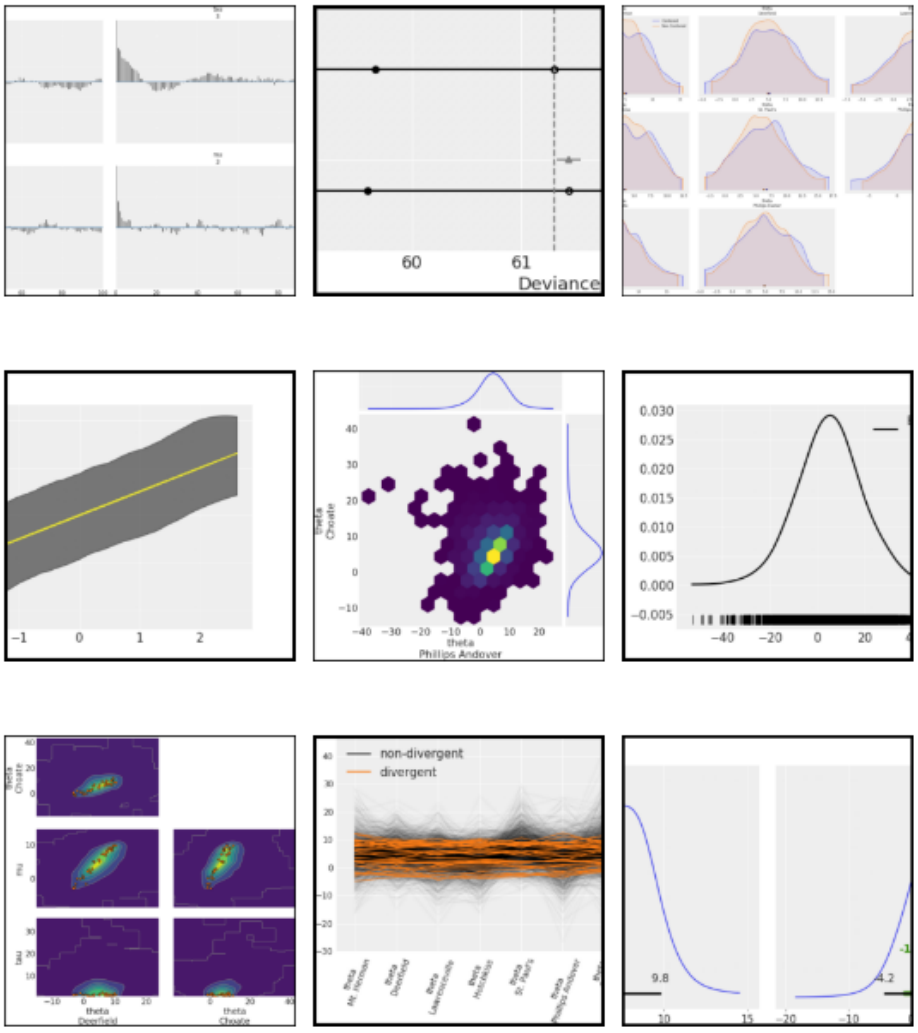

Figure 2: Sample plots from ArviZ gallery

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.

References

Bessiere, Pierre, Emmanuel Mazer, Juan Manuel Ahuactzin, and Kamel Mekhnacha. 2013. *Bayesian Programming*. 1 edition. Boca Raton: Chapman and Hall/CRC.

Bingham, Eli, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. 2018. "Pyro: Deep Universal Probabilistic Programming." *Journal of Machine Learning Research*.

Brown, Stewart A., Mike Folk, Gregory Goucher, Russ Rew, and Paul F. Dubois. 1993. "Software for Portable Scientific Data Management." *Computers in Physics* 7 (3): 304–8. doi:10.1063/1.4823180.

Daniel Roy. 2015. "Probabilistic Programming." *Http://Probabilistic-Programming.org/Wiki/Home*. http://probabilistic-programming.org/wiki/Home.

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

Foreman-Mackey, D., D. W. Hogg, D. Lang, and J. Goodman. 2013. "Emcee: The MCMC Hammer." *PASP* 125: 306–12. doi:10.1086/670067.

Gabry, Jonah, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. 2017. "Visualization in Bayesian Workflow." *arXiv:1709.01449 [Stat]*, September. http://arxiv.org/abs/1709.01449.

Ghahramani, Zoubin. 2015. "Probabilistic Machine Learning and Artificial Intelligence." *Nature* 521 (7553): 452–59. doi:10.1038/nature14541.

Hoyer, Stephan, and Joe Hamman. 2017. "Xarray: N-D Labeled Ar-

rays and Datasets in Python." *Journal of Open Research Software* 5 (1). doi:10.5334/jors.148.

Rew, R., and G. Davis. 1990. "NetCDF: An Interface for Scientific Data Access." *IEEE Computer Graphics and Applications* 10 (4): 76–82. doi:10.1109/38.56302.

Salvatier, John, Thomas V. Wiecki, and Christopher Fonnesbeck. 2016. "Probabilistic Programming in Python Using PyMC3." *PeerJ Computer Science* 2 (April): 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, Dustin, Matthew D. Hoffman, Rif A. Saurous, Eugene Brevdo, Kevin Murphy, and David M. Blei. 2017. "Deep Probabilistic Programming." *arXiv:1701.03757 [Cs, Stat]*, January. http://arxiv.org/abs/1701.03757.

Tran, Dustin, Alp Kucukelbir, Adji B. Dieng, Maja Rudolph, Dawen Liang, and David M. Blei. 2016. "Edward: A Library for Probabilistic Modeling, Inference, and Criticism." *arXiv:1610.09787 [Cs, Stat]*, October. http://arxiv.org/abs/1610.09787.

Vehtari, Aki, Andrew Gelman, and Jonah Gabry. 2015. "Practical Bayesian Model Evaluation Using Leave-One-Out Cross-Validation and WAIC." *arXiv:1507.04544 [Stat]*, July. http://arxiv.org/abs/1507.04544.

Watanabe, Sumio. 2013. "A Widely Applicable Bayesian Information Criterion." *Journal of Machine Learning Research* 14 (March): 867–97.