The Journal of Open Source Software

PreliZ: A tool-box for prior elicitation

Alejandro Icazatti [1], **Oriol Abril-Pla** [2,3], **Arto Klami** [3], **and Osvaldo A Martin** [1¶]

1 IMASL-CONICET. Universidad Nacional de San Luis. San Luis, Argentina **2** Independent Researcher, Spain **3** Department of Computer Science, University of Helsinki, Finland ¶ Corresponding author

DOI: 10.21105/joss.05499

Software

- Review
- Repository
- Archive

Editor: Olexandr Konovalov

Reviewers:

- @jungtaekkim
- @djmannion

Submitted: 01 March 2023
Published: 22 September 2023

Summary

In a Bayesian modeling workflow, a prior distribution can be chosen in different ways as long as it captures the uncertainty about model parameters prior to observing any data. Particularly, prior elicitation refers to the process of transforming the knowledge of a particular domain into well-defined probability distributions. Here we introduce PreliZ, a Python package aimed at helping practitioners to choose prior distributions.

Statement of need

Specifying useful priors is a central aspect of Bayesian statistics (Gelman et al., 2013; Martin et al., 2021; Martin & Teste, 2022), yet prior elicitation techniques that would make this step easier and more systematic are not routinely used within practical Bayesian workflows (Mikkola et al., 2023). Instead, practitioners typically rely on ad hoc procedures based on a mix of their own experience and recommendations available in the literature, which in general has not been systematized (Sarma & Kay, 2020). One reason is that current solutions are simply not sufficient for practical data analysis and do not integrate well with probabilistic programming libraries (Mikkola et al., 2023). PreliZ is a library for prior elicitation that aims to facilitate the task for practitioners by offering a set of tools for the various facets of prior elicitation. It covers a range of methods, from unidimensional prior elicitation on the parameter space to predictive elicitation on the observed space. The goal is to be compatible with probabilistic programming languages in the Python ecosystem like PyMC and PyStan, which involves including the distributions and their parameterization that are supported by these languages.

Software API and features

PreliZ provides functionality for four core aspects: (1) Specification and manipulation of prior distributions, (2) Visualization of priors and the induced predictive distributions, (3) User interaction components, and (4) Algorithms learning priors from user-provided information. PreliZ distributions are built on top of SciPy distributions, providing easy access to all standard functionality, in particular random sampling and quantiles needed in the elicitation process. However, PreliZ extends many of the distributions for increased user-friendliness. For instance, PreliZ uses a canonical parametrization, instead of loc-scale style used by many of the SciPy distributions, and adds common alternative parametrizations, e.g. Beta-distribution also in terms of mean and standard deviation. Finally, PreliZ also includes some distributions not available in SciPy, like the HalfStudentT, or LogitNormal.

Preliz provides a range of plotting commands, including probability densities and cumulative densities (`plot_pdf` and `plot_cdf`) as well as extensive numerical summaries covering e.g. equal tailed intervals and highest density intervals. All plots support easy integration of input

functionality, with the method `plot_interactive` automatically creating ipywidgets sliders for effortless exploration of how the parameters affect the distribution (see Figure 1).

```
pz.Gamma(mu=2, sigma=1).plot_interactive()
```

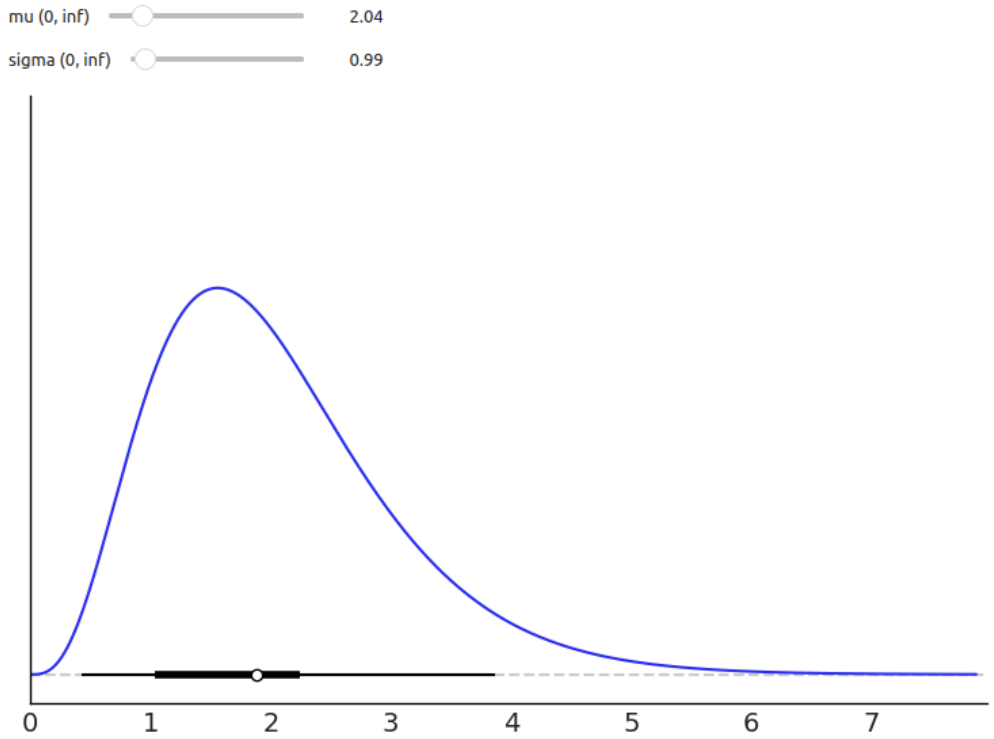

Figure 1: Example of `plot_interactive()` output for Gamma distribution with mu=2 and sigma=1.

PreliZ includes a few algorithms for inferring the suitable prior from the user-provided information, and provides a general API that makes adding new algorithms easy. As a practical example, `pz.maxent(pz.Gamma(mu=4), 1, 10, 0.9)` would find the maximum-entropy Gamma distribution that has 90% of its mass between 1 and 10, with the additional constraint of mean being 4. The library currently implements also the roulette method of Morris et al. (2014), allowing users to find a prior distribution by drawing a histogram (see Figure 2).

```
%matplotlib widget
pz.roulette()
```

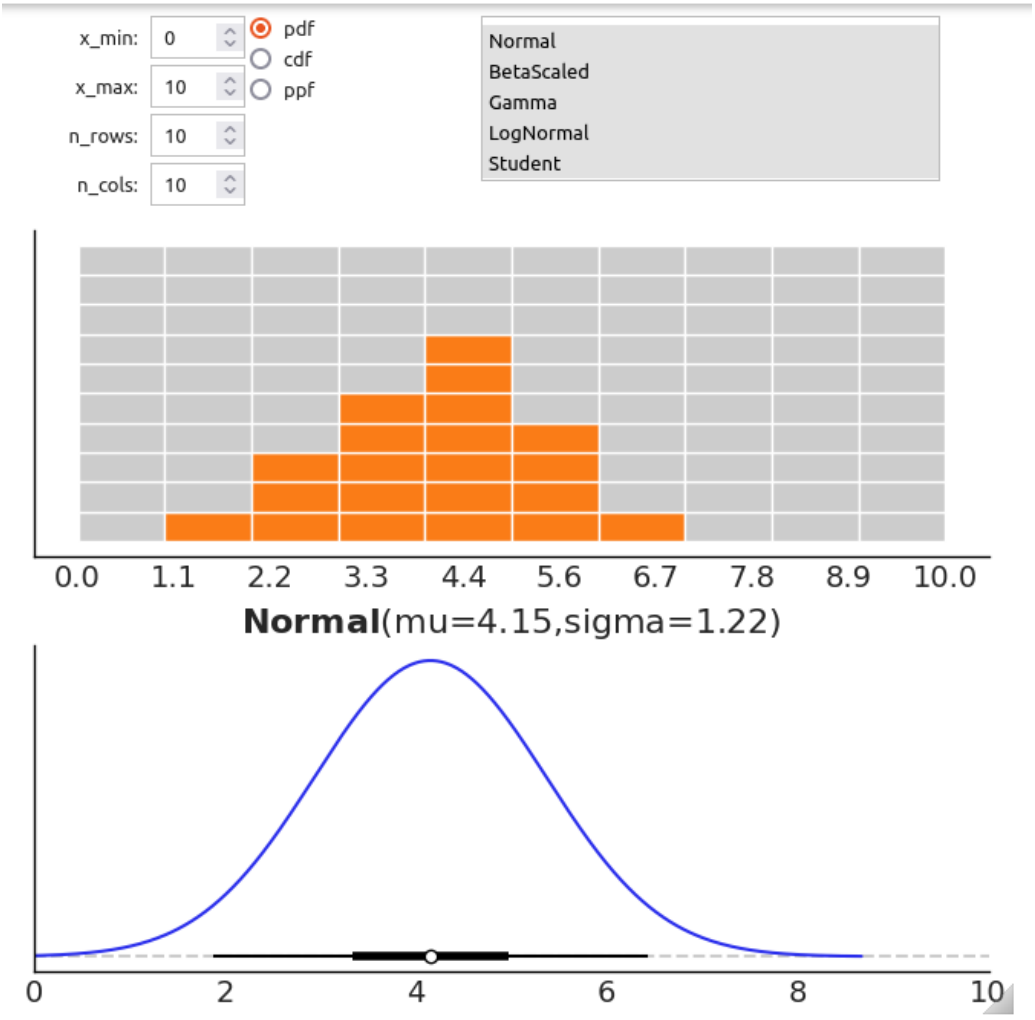

Figure 2: Roulette method.

One of the main goals of PreliZ is to facilitate predictive elicitation, where information provided in the space of observable quantities is converted into a valid prior by a suitable learning algorithm, avoiding the need to provide direct information on the model parameters. PreliZ already implements functionality for this. For instance, `pz.predictive_sliders` (see Figure 3) makes it easy to explore how the prior predictive distribution of an arbitrary probabilistic model changes with the prior.

```
def a_preliz_model(a_mu, a_sigma):
    a = pz.Normal(a_mu, a_sigma).rvs()
    b = pz.Normal(a, 1).rvs(100)
    return b

pz.predictive_sliders(a_preliz_model, samples=50, kind_plot='kde')
```

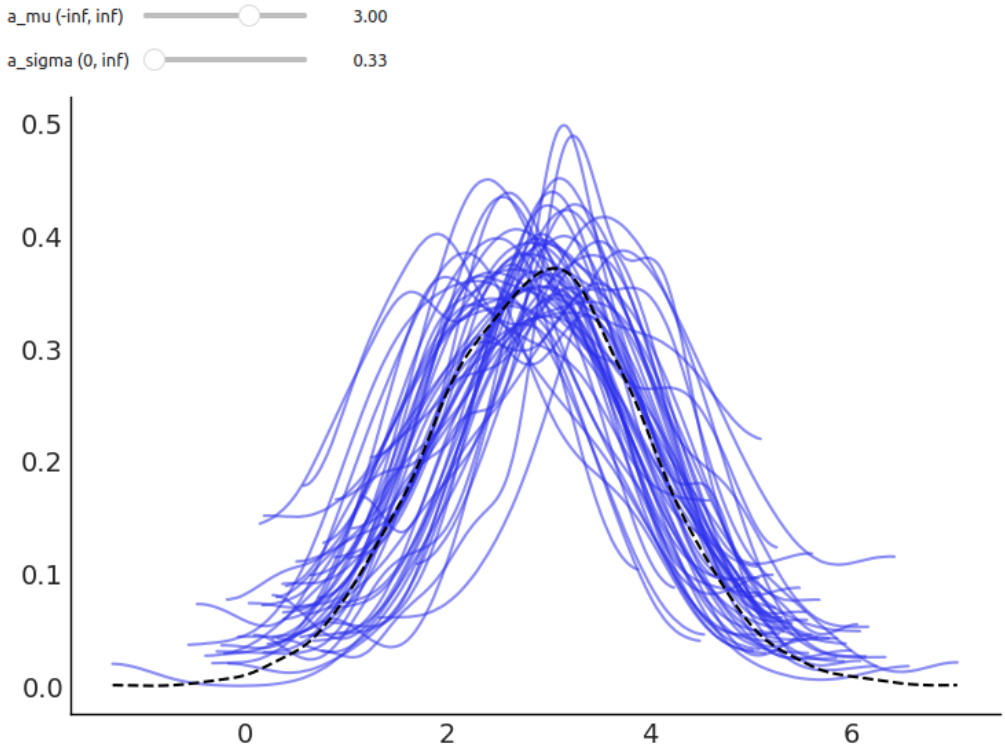

Figure 3: Example of use of `pz.predictive_sliders` for a simple model.

Another function for predictive elicitation is `pz.ppa` (see Figure 4). This is an experimental prior predictive assistant that computes priors consistent with user-made choice of prior predictive samples.

```
def another_preliz_model():
    a = pz.Normal(0, 10).rvs()
    b = pz.HalfNormal(10).rvs()
    y = pz.Normal(a, b).rvs(100)
    return a, b, y

%matplotlib widget
pz.ppa(another_preliz_model)
```

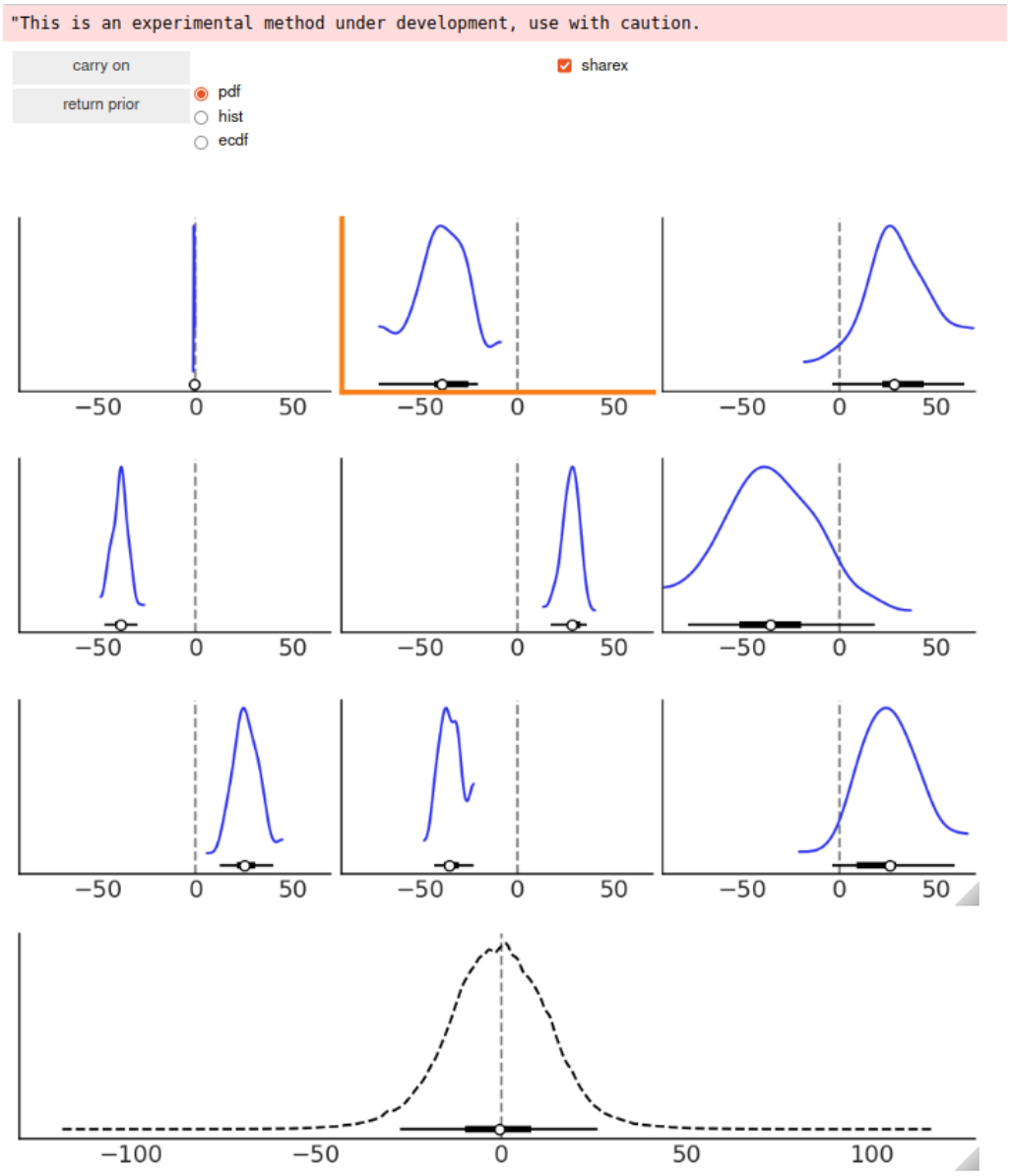

Figure 4: Snapshot of a step of `pz.ppa`.

Software citations

PreliZ is written in Python 3.8+ and uses the following software packages:

- ArviZ (Kumar et al., 2019)
- ipympl (*Ipympl*, 2019)
- ipywidgets (Jupyter widgets community, 2015)
- Matplotlib (Hunter, 2007)
- NumPy (Harris et al., 2020)
- SciPy (Virtanen et al., 2020)

Acknowledgements

We thank our fiscal sponsor NumFOCUS, a nonprofit 501(c)(3) public charity, for their operational and financial support.

This research was supported by:

- the Research Council of Finland Flagship Programme "Finnish Center for Artificial Intelligence" (FCAI)
- the National Agency of Scientific and Technological Promotion (ANPCyT), Grant PICT-02212 (O.A.M.)
- the National Scientific and Technical Research Council (CONICET), Grant PIP-0087 (O.A.M).

References

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian Data Analysis, Third Edition* (3 edition). Chapman; Hall/CRC. https://doi.org/10.1201/b16018

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*, 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Ipympl. (2019). Matplotlib Developers. https://github.com/matplotlib/ipympl

Jupyter widgets community. (2015). *Ipywidgets, a GitHub repository*. https://github.com/jupyter-widgets/ipywidgets

Kumar, R., Carroll, C., Hartikainen, A., & Martin, O. (2019). ArviZ a unified library for exploratory analysis of bayesian models in python. *Journal of Open Source Software*, *4*(33), 1143. https://doi.org/10.21105/joss.01143

Martin, O. A., Kumar, R., & Lao, J. (2021). *Bayesian Modeling and Computation in Python*. https://doi.org/10.1201/9781003019169

Martin, O. A., & Teste, F. P. (2022). A call for changing data analysis practices: From philosophy and comprehensive reporting to modeling approaches and back. *Plant and Soil*. https://doi.org/10.1007/s11104-022-05329-0

Mikkola, P., Martin, O. A., Chandramouli, S., Hartmann, M., Pla, O. A., Thomas, O., Pesonen, H., Corander, J., Vehtari, A., Kaski, S., Bürkner, P.-C., & Klami, A. (2023). Prior Knowledge Elicitation: The Past, Present, and Future. *Bayesian Analysis*, 1–33. https://doi.org/10.1214/23-BA1381

Morris, D. E., Oakley, J. E., & Crowe, J. A. (2014). A web-based tool for eliciting probability distributions from experts. *Environmental Modelling & Software*, *52*, 1–4. https://doi.org/10.1016/j.envsoft.2013.10.010

Sarma, A., & Kay, M. (2020). Prior Setting in Practice: Strategies and Rationales Used in Choosing Prior Distributions for Bayesian Analysis. *Conference on Human Factors in Computing Systems*, 1–12. https://doi.org/10.1145/3313831.3376377

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson,

J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2