pygtc: beautiful parameter covariance plots (aka. Giant Triangle Confusograms)

Sebastian Bocquet

Argonne National Laboratory

Faustin W. Carter

Argonne National Laboratory

27 July 2016

Summary

Multi-dimensional model parameter spaces are commonly sampled using Markov Chain Monte-Carlo (MCMC) methods or more advanced algorithms as implemented, for instance, in `emcee` (Foreman-Mackey et al. 2013) or `PyMultiNest` (Buchner et al. 2014). The recovered parameter constraints are usually displayed on a grid in which the diagonal shows the 1-dimensional posteriors and the lower-left half shows the pairwise projections. Due to the triangular appearance, such plots are typically referred to as "triangle" or "corner" plots. If the parameter space is large, the resulting plot can be visually overwhelming; we refer to such a figure as a Giant Triangle Confusogram (GTC).

Several packages exist to produce these plots and range in functionality from the lightweight `corner` (Foreman-Mackey 2016), which can display a single set of likelihood surfaces from a single analysis, to the feature-heavy `getdist` (Lewis 2015), which allows arbitrary large data sets and contains nearly every conceivable option and customization (at the expense of a steep learning curve). `pygtc` aims to fill the parameter space in between by producing beautiful publication-ready figures with as little as a single line of code, while allowing for an arbitrary number of sets of likelihood surfaces (a crucial feature in our field of cosmology). Various user-friendly options allow for further adjustments and ensure that details such as font sizes are chosen to match the layout of several astrophysical journals.

Figure 1 shows an example figure (comparing three sets of data from different analyses) that was produced with the `pygtc` package and formatted for the Astrophysical Journal, all with the following line of code:

```
GTC = pygtc.plotGTC(chains=[dat1,dat2,dat3],
                    chainLabels=["data1", "data2", "data3"],
                    paramNames=['foo', 'bar', '$C$', '$\\nu$'],
                    truths=(None, 1, None, 0),
                    truthLabels='ref',
                    figureSize='APJ_page')
```

`pygtc` relies on `matplotlib` (Hunter 2007) and `scipy` (Jones et al. 2001–2001--) to produce aesthetically appealing GTCs, and is designed to accept the standard output of the `emcee` package or any other likelihood
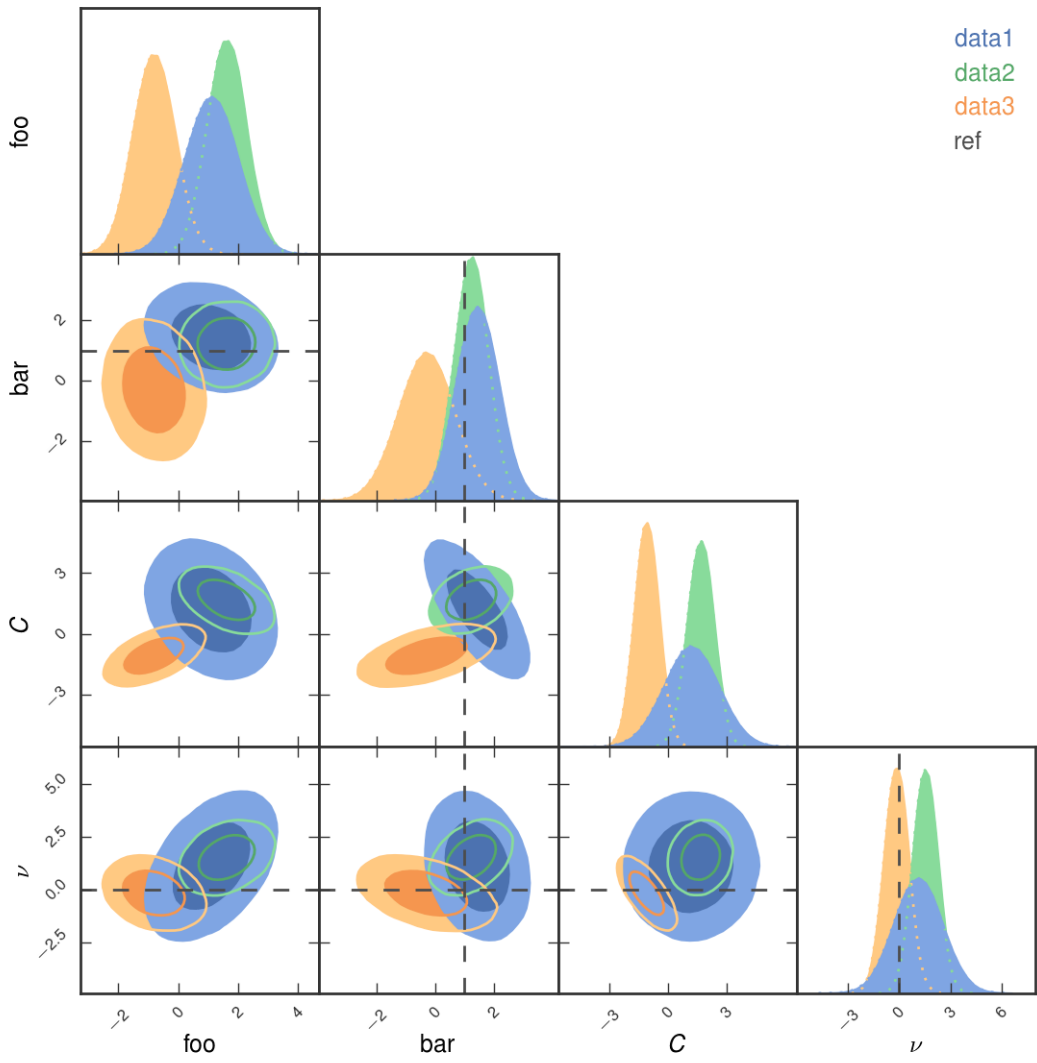
Figure 1: A GTC produced by *pygtc*.

sampler that returns a list of (weighted or unit-weighted) sample points. The source code for `pygtc` is available at Zenodo (Bocquet and Carter 2016b) and development is ongoing at GitHub (Bocquet and Carter 2016a).

References

Bocquet, Sebastian, and Faustin W. Carter. 2016a. "Pygtc GitHub Repository." https://github.com/SebastianBocquet/pygtc.

————. 2016b. "Pygtc Zenodo Archive." doi:to be determined.

Buchner, J., A. Georgakakis, K. Nandra, L. Hsu, C. Rangel, M. Brightman, A. Merloni, M. Salvato, J. Donley, and D. Kocevski. 2014. "X-ray spectral modelling of the AGN obscuring region in the CDFS: Bayesian model selection and catalogue" 564 (April): A125. doi:10.1051/0004-6361/201322971.

Foreman-Mackey, Daniel. 2016. "Corner.py: Scatterplot Matrices in Python." *The Journal of Open Source Software* 24. doi:10.21105/joss.00024.

Foreman-Mackey, Daniel, David W. Hogg, Dustin Lang, and Jonathan Goodman. 2013. "Emcee: The MCMC Hammer." *Publications of the Astronomical Society of the Pacific* 125 (925): 306. http://stacks.iop.org/1538-3873/125/i=925/a=306.

Hunter, J. D. 2007. "Matplotlib: A 2D Graphics Environment." *Computing In Science & Engineering* 9 (3). IEEE COMPUTER SOC: 90–95.

Jones, Eric, Travis Oliphant, Pearu Peterson, and others. 2001–2001--. "SciPy: Open Source Scientific Tools for Python." http://www.scipy.org/.

Lewis, Antony. 2015. "Getdist GitHub Repository." https://github.com/cmbant/getdist.