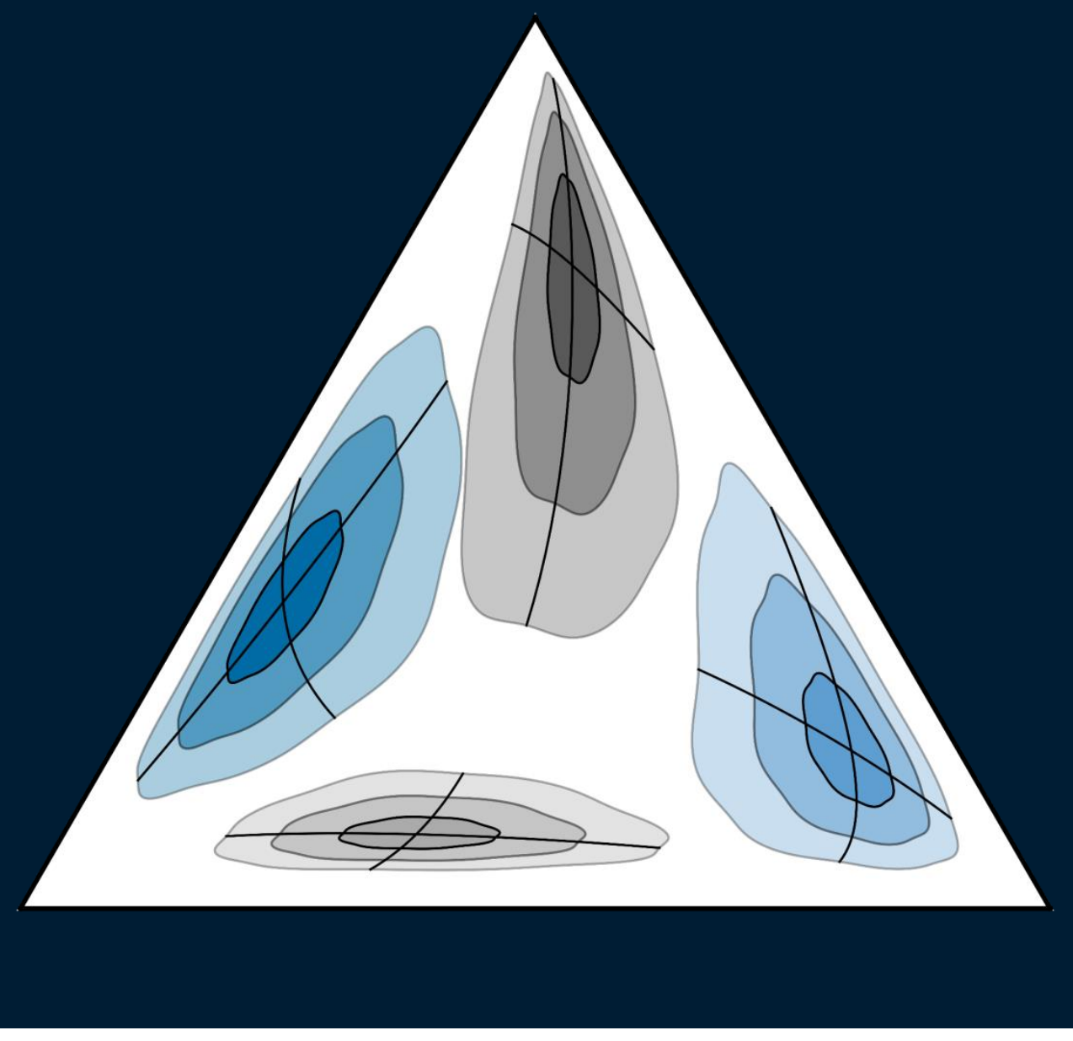
Geochemical Data Analysis Workflows with pyrolite

Supporting reproducible programmatic approaches to geochemical data workflows with community-driven open-source tools.

Morgan Williams and Louise Schoneveld (CSIRO Mineral Resources)



- Data-driven approaches to geochemical problems are becoming much more feasible with increasingly large volumes of data
- Programmatic approaches offer increased reproducibility for geochemical data workflows
- *pyrolite* is an open source Python package for working with geochemical data; here we illustrate its use in some geochemical data workflows

MORE INFORMATION
Morgan Williams
Mineral Resources
morgan.williams@csiro.au
@metasomite

Towards Transparent and Reproducible Geochemical Data Analysis

Geochemists increasingly have access to advanced instrumentation enabling rapid acquisition of multivariate analytical datasets. However, the software tools we use to reduce and interrogate this data have not seen the same degree of evolution as our instruments. To extract the most from large multivariate geochemical datasets, we can make use of modern tools and statistical analysis methods, and in some cases leverage machine learning workflows. Critically, we should strive to develop robust and reproducible data analysis workflows in the same way we strive to ensure reproducible analyses of our samples.

An emerging shift towards the use of high-level programming languages such as Python, R and Julia provides numerous opportunities to increase the reproducibility of data reduction, analysis and visualisation processes, and add further value to our science through the development of software tools. In particular, a declarative programmatic approach to data processing and analysis makes these tasks more repeatable, and actively encourages documenting the process. When combined with versioning and environment management, this approach enables reproducible workflows which can be shared such that others can effectively examine, compare and re-use existing code as needed.

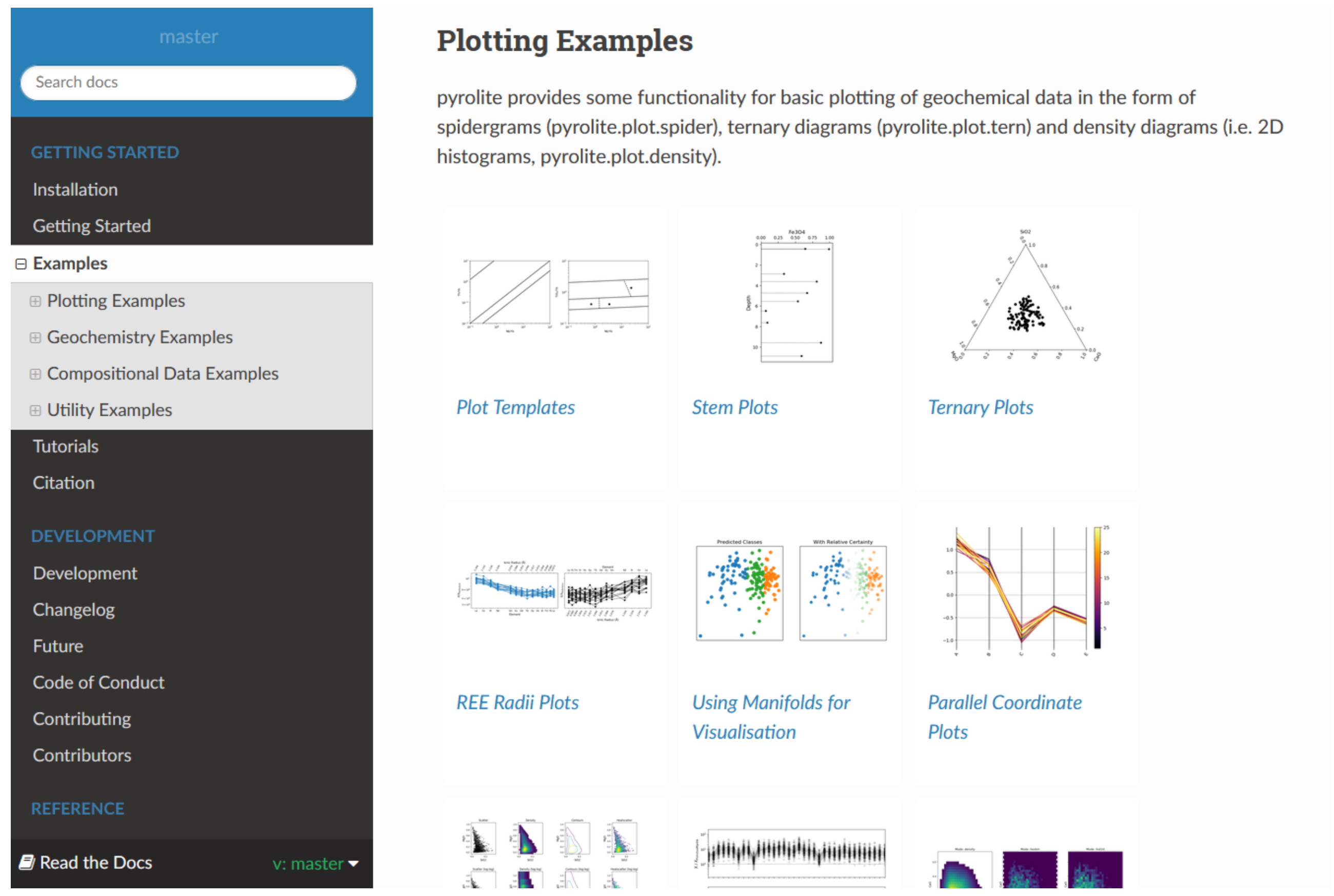
To minimise the activation energy required for adoption of programmatic approaches, domain-focused tools with 'batteries included' functionality are needed such that new users can get off the ground quickly. In developing *pyrolite*, we hope to address the relative scarcity of geochemistry focused tools, and enable geochemists to access the flexibility and analytical power that Python provides.

pyrolite

pyrolite is an open source Python package for transforming, analysing and visualising geochemical and compositional data. The package contains a suite of functions commonly used in geochemical data workflows, including log-transforms for working with compositional data in a robust manner, scaling between units and simple element-oxide conversion. pyrolite also implements several common geochemical visualisations and plot templates, and provides easy access to data-density based visualisation methods better adapted to large multivariate datasets with hundreds to hundreds of thousands of samples. Beyond providing foundational functionality, pyrolite also provides a framework to encode and document relevant algorithms recently introduced to the geochemistry community (e.g. lambdas for parameterising rare earth element profiles[1], bootstrap resampling methods[2]) and also link geochemical data to common machine learning frameworks (e.g. scikit-learn[3]). The package and related tools continue to be actively developed, and the package has recently been peer reviewed and published[4].

Documentation

The documentation for pyrolite has been a core focus alongside the package itself. The current documentation hosted at pyrolite.readthedocs.io includes installation instructions, a range of examples together with extensive API documentation describing the use of each function and class.



Plotting Examples

pyrolite provides some functionality for basic plotting of geochemical data in the form of spidergrams (pyrolite.plot.spider), ternary diagrams (pyrolite.plot.tern) and density diagrams (i.e. 2D histograms, pyrolite.plot.density).

Plot Templates · *Stem Plots* · *Ternary Plots* · *REE Radii Plots* · *Using Manifolds for Visualisation* · *Parallel Coordinate Plots*

Figure 1: Screenshot of the examples galleries hosted within the documentation site.

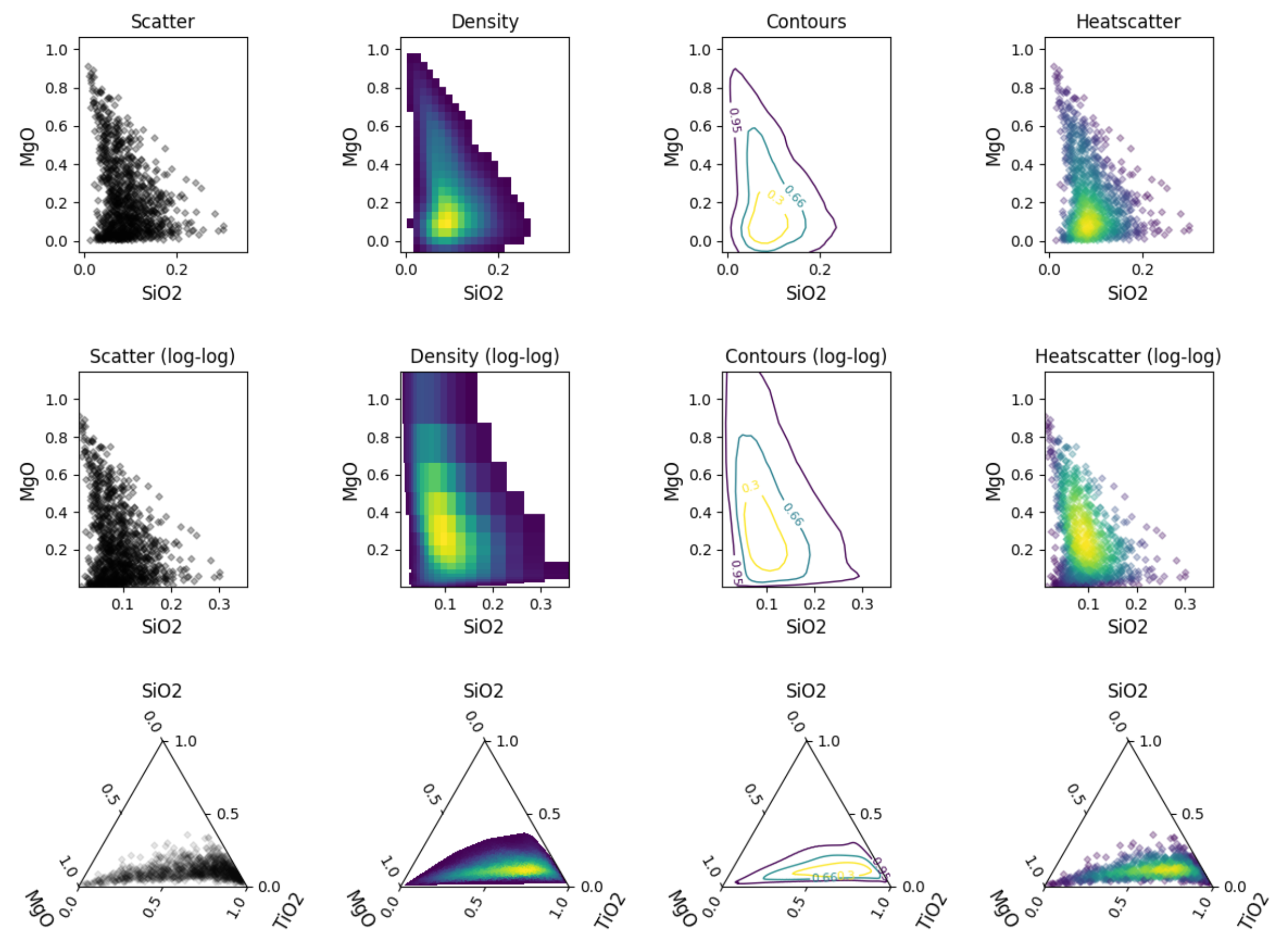


Figure 2: Illustration of some common data-density based visualisations generated by pyrolite, highlighting the effect of overplotting (left) and how other methods of representing the summary information of a data distribution can be used to retain key information. Each of these visualisations are useful in different scenarios. For example, the heatscatter diagram allows data density to be summarized and point-location information to be retained - but could become complicated if more than one distribution is shown on a single set of axes.

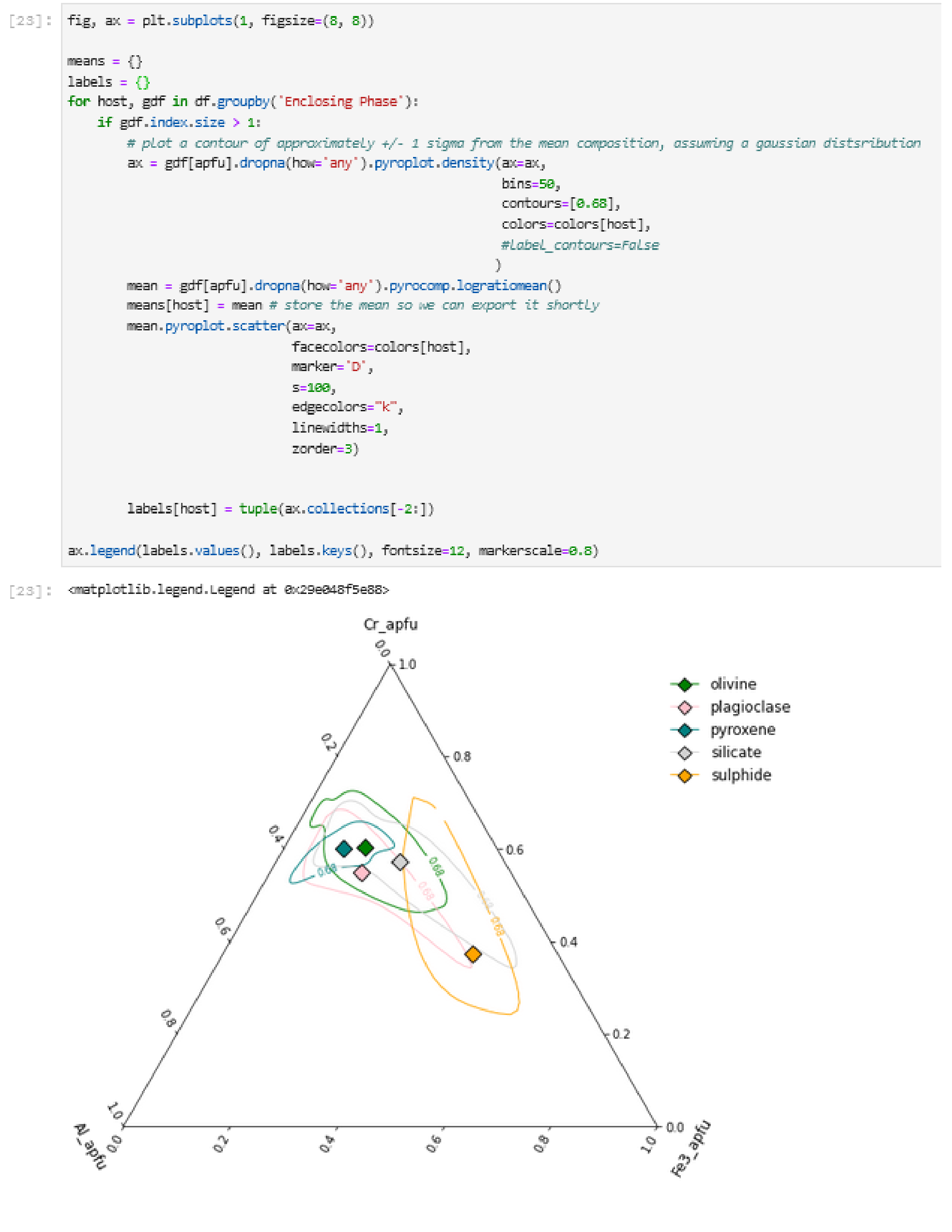
Get Involved!

pyrolite aims to be developed for the geochemistry community, and by the geochemistry community.

We hope to foster a growing community of users and contributors to ensure the long-term sustainability and usefulness of the project. All forms of contribution to the project are welcome, from identifying issues, requesting features, contributions to documentation and the code itself. Contributions are also acknowledged in the documentation, and where possible in publications focusing on the package itself. A community forum is provided on Gitter, enabling a direct link to the developers for general questions and debugging. Feel free to get in contact if you'd like to know more.

Workflow Examples

Together with this poster, we have put together a series of examples in Jupyter notebooks illustrating different aspects of geochemical data workflows in which pyrolite can prove useful. These include sections focusing on compositional data, transforming geochemical datasets, and connecting geochemical data to machine learning pipelines. These notebooks are hosted on GitHub, and you can access and run them directly using Binder.



```
[23]: fig, ax = plt.subplots(1, figsize=(8, 8))

      means = {}
      labels = {}
      for host, gdf in df.groupby('Enclosing Phase'):
          if gdf.index.size > 1:
              # plot a contour of approximately +/- 1 sigma from the mean composition, assuming a gaussian distsribution
              ax = gdf[apfu].dropna(how='any').pyroplot.density(ax=ax,
                                                                bins=50,
                                                                contours=[0.68],
                                                                colors=colors[host],
                                                                #label_contours=False
                                                               )
              mean = gdf[apfu].dropna(how='any').pyrocomp.logratiomean()
              means[host] = mean # store the mean so we can export it shortly
              mean.pyroplot.scatter(ax=ax,
                                    facecolors=colors[host],
                                    marker='D',
                                    s=100,
                                    edgecolors="k",
                                    linewidths=1,
                                    zorder=3)


              labels[host] = tuple(ax.collections[-2:])

      ax.legend(labels.values(), labels.keys(), fontsize=12, markerscale=0.8)

[23]: <matplotlib.legend.Legend at 0x29e048f5e88>
```

Figure 3: A section of one Jupyter notebook workflow example, here examining the geochemical features of spinels found within different host minerals[5]. This notebook includes sections on examining the nature of compositional data, calculating unbiased means and examining multivariate compositional data distributions (such as those indicated by the data density contours here).

As Australia's national science agency and innovation catalyst, CSIRO is solving the greatest challenges through innovative science and technology.

CSIRO. Unlocking a better future for everyone.

REFERENCES

[1]: O'Neill, H.S.C. (2016). The Smoothness and Shapes of Chondrite-normalized Rare Earth Element Patterns in Basalts. J Petrology 57, 1463–1508. https://doi.org/10.1093/petrology/egw047; [2]: Keller, C.B. and Schoene, B. (2012). Statistical geochemistry reveals disruption in secular lithospheric evolution about 2.5 Gyr ago. Nature 485, 490–493. https://doi.org/10.1038/nature11024; [3]: Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, and M., Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825−2830. [4]: Williams, M.J., Schoneveld, L., Mao, Y., Klump, J., Gosses, J., Dalton, H., Bath, A. and Barnes, S. (2020). pyrolite: Python for geochemistry. Journal of Open Source Software 5, 2314. https://doi.org/10.21105/joss.02314; [5]: Schoneveld, L., Barnes, S.J., Williams, M., Vaillant, M.L. and Paterson, D. (2020). Silicate and Oxide Mineral Chemistry and Textures of the Norilsk-Talnakh Ni-Cu-Platinum Group Element Ore-Bearing Intrusions. Economic Geology. https://doi.org/10.5382/econgeo.4747

ACKNOWLEDGEMENTS

The authors would like to thank all of those who have contributed to pyrolite thus far, and those that have encouraged its further development.