SciPy: Scientific Computing in Python

Overview

SciPy is a Python package built on the NumPy stack that provides users with functionality for use in scientific computation. Modules included focus on topics such as: optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers, and other common scientific and engineering tasks. SciPy leverages the array interfacing made standard by NumPy and utilized in other packages such as matplotlib, pandas, scikit-learn, and others. It is developed in an open source fashion and distributed under a BSD-3 license. In this report, I will focus on the recent changes made by the community as well as the general decision making process involved in implementing changes.

Decision Process

SciPy originally operated under the leadership of a team of Core Developers. These were founders and major stakeholders in the project who lead the general vision and direction of SciPy's early development. As the community and project became larger (sometime around 2017), SciPy Core Developers decided to adopt a more formal governance model based largely off of Jupyter/IPython and NumPy governance.

The Core Developers decided on a "Benevolent Dictator for Life" (BDFL) style model augmented with a Steering Council. The current BDFL is Pauli Virtanen. Officially, the BDFL has the ultimate authority on all decisions made within the project. In practice, the BDFL rarely uses this authority and allows almost all operations to be conducted by the Steering Council and the community of contributors. To maintain the benevolence of the BDFL, the community is encouraged to regularly fork the main project and develop it as they see fit, especially if decisions are not in line with their view. The BDFL is appointed for life (as in the name), but may step down at any time and is also allowed to appoint their own successor. According to the governance policy, the intention of the BDFL is to act more as a fallback decision maker rather than a CEO or development lead.

The Steering Council consists of major members of the contributing community, and originally consisted of the Core Developers at the time of the new governance model adoption. To be considered for membership on the Council, contributors must have substantial contributions in either quality or quantity and must have maintained their contributions for at least 1 year. They must be nominated and voted in by the current council members. Membership consideration is not based off of any specific metric in an effort to avoid people gaming the system by focusing on such metrics. Any member can be voted off of the Council by the other members.

At the lead of the Steering Council is a Council Chair. The Chair is appointed by the other members of the Council and is appointed indefinitely. The Chair is responsible

for organizing and publishing a bi-yearly review (mid-April and mid-October) through the SciPy Roadmap document. They must also ensure that any private communications among the council members is published in an accessible way to the contributing community. This is to maintain an open and transparent leadership. The Chair also ensures that the Council membership is current and that no members have been inactive for more than 1 year. Finally, they maintain and publish all organization and policy documentation such as the governance policy.

The Steering Council also appoints a Release Manager who serves for one or two version releases. The Release Manager is in charge of the content and timing of version releases as well as the creation of the release and subsequent announcement to the community.

Should the council need to vote, they use an Apache Foundation style voting process. This means that for votes count as +1, against votes count as -1 and must include rationale. Members may vote fractionally. This is only to be used in informal voting processes; in the event a more formal vote is required a longer voting window should be opened with time for discussion and participation from all council members.

Similarly to the original Core Developer model, the Steering Council members are the main facilitators of the contribution process. Community development happens on GitHub, although general feature discussion, planning, and development are also organized through the scipy-dev mailing list. Major contributions are expected to be discussed both before and during their development to ensure they align with the community's vision. Many changes are discussed through issues on the SciPy GitHub as well, but are always a more focussed discussion. All non-trivial contributions (trivial being something like a typo) must be submitted through a pull request on GitHub. The PR must then pass automatic testing and go through a review. Major code review may be necessary and can be a big hangup for contributions as it might require a more specialized member of the community to review it.

Several criteria guide reviewers on their analysis of PRs. Is the method applicable to many fields and "generally agreed" to be useful? Does it fit the topic of the submodule, and does it not require an extensive support framework to operate? Does the implementation look sound and unlikely to need much tweaking in the future? Does someone want to contribute it? Does someone want to review it? Obviously the last two questions apply more in the general discussion of feature addition rather than in the review process.

Recent Developments in SciPy

Many documentation fixes have been implemented since the semester has started (including my own addition of examples). Tutorials were updated to maintain consistency with the current codebase. More testing and benchmarking was added as

this is a long term goal of the developers to include benchmarking for every feature in SciPy.

In general, most of the regular contributions are bugfixes, documentation updates, readability improvements, and additions of tests. This is perhaps a great lesson in that most development is incremental and it is rare to see pull requests that add major functionality. This is actually desirable as it allows for the development to be more conservative, better controlled, and certainly more maintainable.

Looking at the SciPy roadmap shows the direction of near future updates. The developers plan to improve the stability of scipy.linalg's use of OpenBLAS as it currently has issues with threading. Also, the developers plan to implement sparse arrays in addition to sparse matrices (i.e. sparse arrays that act like np.ndarray). Other plans are also listed for improvements to FFT and adding support to distributed arrays and GPU arrays.

Conclusion

My report here demonstrates the importance of a structured governance policy and organized documentation in a large project's development. Through the documents published and maintained on SciPy's website and GitHub I was able to navigate their contribution process, gather information about their history, understand their decision making process and leadership model, and ultimately contribute myself. I am proof that this kind of organization is critical in maintaining an active and productive community of contributors.