Support the Python Numerical Core

Joseph Harrington, University of Central Florida, jh@physics.ucf.edu
Ralf Gommers, Quansight, rgommers@quansight.com
Chelle Gentemann, Earth and Space Research, cgentemann@esr.org
Derek Buzasi, Florida Gulf Coast University, dbuzasi@fgcu.edu
Kevin Stevenson, Space Telescope Science Institute, kbs@stsci.edu
Joshua Pepper, Lehigh University, joshua.pepper@lehigh.edu
Perry Greenfield, Space Telescope Science Institute, perry@stsci.edu
Shubham Kanodia, Pennsylvania State University, szk381@psu.edu
Thomas Beatty, University of Arizona, tgbeatty@email.arizona.edu
Ryan Challener, University of Central Florida, rchallen@knights.ucf.edu
Joe Ninan, Pennsylvania State University, jpn23@psu.edu
Jessie Christiansen, Caltech/IPAC-NExScI, jessiec@caltech.edu
Arif Solmaz, Çağ University, arifsolmaz@cag.edu.tr
Erik Tollerud, Space Telescope Science Institute, etollerud@stsci.edu
Nicholas Earl, Space Telescope Science Institute, nearl@stsci.edu
Pey Lian Lim, Space Telescope Science Institute, lim@stsci.edu
Larry Bradley, Space Telescope Science Institute, lbradley@stsci.edu
Elisabeth Newton, Dartmouth College, Elisabeth.R.Newton@dartmouth.edu
Rachel Akeson, Caltech/IPAC, rla@ipac.caltech.edu
Megan Sosey, Space Telescope Science Institute, sosey@stsci.edu
Philip Hodge, Space Telescope Science Institute, hodge@stsci.edu
Paulo Miles-Páez, University of Western Ontario, ppaez@uwo.ca
Kathleen Labrie, Gemini Observatory, klabrie@gemini.edu
Henry Ngo, National Research Council of Canada, Henry.Ngo@nrc-cnrc.gc.ca
Sara Ogaz, Space Telescope Science Institute, ogaz@stsci.edu
Darren Williams, Penn State University, dmw145@psu.edu
Michael Himes, University of Central Florida, mhimes@knights.ucf.edu
Kathleen McIntyre, University of Central Florida, kmcintyre@knights.ucf.edu
Adrienne Dove, University of Central Florida, adrienne.dove@ucf.edu
Joshua Colwell, University of Central Florida, josh@ucf.edu
Joe Llama, Lowell Observatory, joe.llama@lowell.edu
Ryan T. Hamilton, Lowell Observatory, rhamilton@lowell.edu
Geert Barentsen, Bay Area Environmental Research Institute, geert.barentsen@nasa.gov
Ryan Terrien, Carleton College, rterrien@carleton.edu

Type of Activity: Infrastructure Activity

Executive Summary and Recommendations

Open-source software (OSS) promotes reproducibility and efficiency in science. The most popular OSS framework in astrophysics is the Python Numerical Core (PNC), including the NumPy, SciPy, Matplotlib, Pandas, and Scikit-learn packages. With over 5,000,000 users, these projects have grown beyond the volunteer scale and require financial support.

Open-Source Software in Science

Much of the activity in Earth and space science involves crunching numbers on computers, whether in data analysis or theoretical modeling. As calculation complexity has grown, so has the need to share codes rather than writing one's own versions from scratch. For example, few astronomers would think of rewriting the calibration pipeline of a facility telescope such as Hubble, and most users of general circulation models download one of the large, well maintained public codes rather than starting from scratch. Those who do it from scratch typically do so as their career focus. It is becoming recognized that scientific papers cannot adequately describe most data analyses or numerical models sufficiently to reproduce the numbers they report, that the code itself is the ultimate documentation of the calculation, and that therefore it must be disclosed to support scientific claims made from it (Fomel and Claerbout 2009, introduction to *Computing in Science and Engineering* special issue on Reproducible Research).

Exchange of software is difficult if there are components that the recipient cannot run, for example, for lack of a license. Educating students with proprietary software has the disadvantage that they may lose access to the tools they wrote when they leave school. Similarly, professionals changing jobs may leave behind their access to proprietary environments. As OSS solutions respond directly to the needs of the user, not of shareholders or customers in other fields and with different priorities, they have matched or surpassed proprietary tools in essentially every measure, including efficiency, ease of use, documentation, user support, features, robustness, and language quality.

Today, most new investigators learn with OSS tools, many existing projects are converting to OSS, and few projects move from OSS to proprietary software. A recent National Academies study provides detail and numerous white papers supporting OSS in space science (National Academies of Science, Engineering, and Medicine 2018). It calls on NASA to support both the basic OSS packages used in science as well as discipline-specific packages, such as astronomy's AstroPy. This paper outlines the case for the basic packages used in nearly all astrophysics-related research, and the need to fund them.

The Python Numerical Core

The most popular OSS platform for numerical computing, including astrophysics-related work, is the Python language and its Python Numerical Core (PNC). Python was written as a general-purpose, high-level, object-oriented computing language. It was designed for instruction as well as professional use, so it is highly consistent and quite simple; Python code is commonly *shorter* than the pseudocode found in textbooks. Separating the numerical components from the base language has allowed numerical experts to design and maintain those packages. There are many numerical packages, but the five most widely used are the PNC:

- NumPy - the core array object and the most fundamental routines using it (e.g., trigonometry, random numbers, simple statistics)
- SciPy - more advanced or specialized routines using the array object
- Matplotlib - publication-quality 2D and basic 3D plotting and data visualization routines
- Pandas - a framework for structured and unstructured statistical data analysis
- Scikit-learn - machine-learning routines

The web site uniting the numerical Python world is http://scipy.org/ .

Developing, Managing, and Funding the PNC

Each of the PNC projects began and spent many years as a volunteer, "scratch your own itch" project. Some beat stiff competition to gain a large following. Some, such as NumPy, underwent forks, reunifications, and other gyrations before becoming the widely used packages that they are today. Throughout, the developer communities have been drawn from and guided by the user community, through mailing-list discussions and multiple conferences annually, throughout the world.

Today, each package has hundreds of contributors, with many dozens active at any given time. A core group of about ten developers per package are the gatekeepers to the sources, with commit rights. There is formalized governance for major decisions. Some packages have a leader, with ultimate authority and the understanding that it will not be used except to break a consensus deadlock, which is rare; others have a small consensus council. There are detailed roadmaps and planning processes, codes of conduct, deep commitments to testing and documentation, and carefully controlled release cycles. Changes come slowly, after careful consideration and long, open testing periods. Backward-incompatible changes are extremely rare and well heralded through a years-long deprecation process. This makes the software very reliable and stable.

The PNC has had a remarkable uptick in use. Statistics from the GitHub repository put the number of projects with files saying “import numpy” at over 220,000. Many of these are astrophysics repositories, but we believe that most astrophysics codes are not on GitHub. Nearly all high-profile astrophysics projects use the PNC for at least some of their code, and many use it for all their code. These include the LSST, HST, and JWST calibration pipelines, as well as numerous probe data pipelines. Essentially all discipline-specific packages, including AstroPy, depend fundamentally on the PNC packages, and especially NumPy.

The uptick in users has stressed the volunteer community nearly to the breaking point. Each volunteer chooses what to work on, making it difficult to get boring or low-credit tasks done. Such tasks are often critical to users, such as rolling releases, maintaining documentation, answering user questions, maintaining servers, writing tests, porting the software to new hardware, optimizing it for new hardware, managing volunteers, and raising funds and awareness. This work totals about ten full-time equivalent (FTE) employees per project, at this point. Most critical is directing all the work. Much of the work is highly technical, requiring experienced software engineers or numerical-computing-hardware specialists who are not themselves scientists. Many projects are difficult to split into tasks small enough to spread among many part-time volunteers.

To solve these issues, community leaders formed NumFOCUS, a US non-profit that raises funds for member projects and hires developers and others to work on them. NumFOCUS has the legal and financial management team to handle gifts, grants, and contracts. The PNC projects are all members of NumFOCUS, meaning they have made certain governance and management commitments to ensure community control and maintain non-profit status.

Funding Efforts to Date

NumFOCUS is successful only to the extent of the funds it raises. The PNC user community is huge, spanning corporations, non-profit think-tanks, universities, schools, government, and private users worldwide. However, the nature of OSS is that there is no obligation to pay, and people seldom do. The first to step up have been companies, including some founded by PNC leaders and early developers, that use Python to deliver custom software or to analyze data, generally in the commercial world. It is difficult to assess the fraction of use due to astrophysics, but we do know that the PNC underpins a substantial fraction of NASA's and NSF's productivity. If the packages were commercial software bearing license fees comparable to commercial offerings in this space, the income generated would be in the tens of millions of dollars from astrophysics alone, as a commercial software site license for a NASA center is many hundreds of thousand of dollars.

There are infrastructure software development programs at NSF, and we have applied to these. After all, it would disrupt science if the PNC were abandoned as unmaintainable. However, these programs are for one-time development of new software, not for maintaining existing capability. Our proposal was not considered very exciting, and in fact it isn't. Volunteers are eager to do the exciting parts of maintenance. It is the boring-but-necessary tasks for which we seek funds.

Request

It would not be fair to ask NASA and NSF to fund all of the PNC's needs. There is growing commercial use, as well as use in other scientific communities. At the same time, someone needs to step up and make a meaningful financial commitment, or the future of the PNC as maintainable software is in doubt. We feel that 10% of the total need is a fair request for US astronomy sources (i.e., NASA Earth and Space Science and NSF Astronomy and Astrophysics). This is just for the PNC. Funding astronomical tools, such as AstroPy, is important, but separate.

With five projects and ten FTE per project, the request comes to 50 FTE. At an average senior developer salary of $120,000, this is $6M annually, plus burdens and expenses (conferences, servers, etc.). NASA's share would be ~$1-2M/year, in very rough numbers. This is a very small cost for NASA, more substantial for NSF, but easily carried by both, given the scope of their operations and the benefit to the community. A fair split between the two could go according to the sizes of their research and analysis programs. Additional contributions should also come from NASA's non-science components and non-astronomy programs at NSF.

There are several mechanisms that could be used. The simplest would simply be a contract with NumFOCUS, renewed annually, with the deliverable being a report on activities and the penetration of the software. A more market-based approach would be to require grant recipients using primarily OSS solutions to identify what they used, and to allocate the equivalent of a few individual commercial software licenses per grant to those projects. We strongly suspect this would represent much more money, and it would be more fair, ensuring all relevant projects received support. However, it would be a major administrative and bookkeeping headache for all concerned.

Regardless of how it is done, funding the PNC is critical. Thousands of NASA- and NSF-funded research projects depend on it. Major NASA facilities and missions depend on it. Numerous courses at colleges and universities, and even some high schools, use it. As the number of users shoots past the five million mark, managing the hundreds of volunteer contributors, supporting the community, and maintaining the core activities and infrastructure can only be done by full-time, paid professionals. If we do not fund the PNC, the software will stagnate and grow unmanageable. Documentation will go out of date, new hardware will not be supported, and servers not be consistently available. The quality and frequency of releases will not be able to be maintained. User questions will not be answered. This will happen slowly, and as users find that the PNC does not meet their needs, they will switch to alternatives. Switching a large toolset to another language can take years. Finding those alternatives not quite up to par, they will sink time into them, further reducing their efficiency. The large number of alternatives will dilute effort, ensuring that none of them becomes what the PNC is now. It will be more difficult to share software if it is not based on a widely accepted foundation like the PNC.

We have already spent the effort to build the PNC past the equivalent commercial offerings. Making and keeping it available to a large and fast-growing user base is a different story. It is a worthy project, critical to NASA and NSF. We ask that Astro2020 specifically call out the PNC for support at the ~$1-2M level annually.

References

National Academies of Science, Engineering, and Medicine 2018. *Open Source Software Policy Options for NASA Earth and Space Sciences.* National Academies Press. https://www.nap.edu/read/25217

Fomel, S., and J. Claerbout 2009. Reproducible Research in *Computing in Science & Engineering* 11, 5. Introduction to special issue on Reproducible Research. https://doi.org/10.1109/MCSE.2009.14