Progress Report: NumPy and SciPy Documentation in 2009

Joseph Harrington (jh@physics.ucf.edu) – *U. Central Florida, Orlando, Florida* USA
David Goldsmith (dgoldsmith_89@alumni.brown.edu) – *U. Central Florida, Orlando, Florida* USA

In the Spring of 2008, the SciPy Documentation Project began to write documentation for NumPy and SciPy. Starting from 8658 words, the NumPy reference pages now have over 110,000 words, producing an 884-page PDF document. These pages need to undergo both presentation and technical review. Our plans include SciPy documentation as well as several user guides, tutorials, and pamphlets. A critical need at this point is a stable funding stream to support the Documentation Coordinators who have been instrumental to the success of this project.

Abstract

In the Spring of 2008, the SciPy Documentation Project began to write documentation for NumPy and SciPy. Starting from 8658 words, the NumPy reference pages now have over 110,000 words, producing an 884-page PDF document. These pages need to undergo both presentation and technical review. Our plans include SciPy documentation as well as several user guides, tutorials, and pamphlets. A critical need at this point is a stable funding stream to support the Documentation Coordinators who have been instrumental to the success of this project.

Introduction

We maintain that, under any good development model, a software project's reference documentation is written approximately concurrently with code. In other words, the software includes full reference documentation as part of each release.

In our experience, many research codes do not follow such a model. Research codes typically start as solutions to specific needs and are written by the researchers who will use them. Documentation is often brief and incomplete, as the small circle of users is generally familiar with the code, having written it or having received training from those who did. The code may even be short enough and the users expert enough that reading the source code is a viable documentation method for them. Most such codes reach end-of-life when the original authors either move on or can no longer maintain them, or when the efforts for which they were written end.

However, some codes achieve a life of their own, with new people taking on maintenance and development in a community open-source effort. If the lack of documentation is not recognized and remedied at this stage, and if the developers do not require documentation to be written concurrently with any new code, the deferred documentation effort can become larger than any reasonable investment of the developers' time.

This was the case with NumPy. Between its forks, rewrites, and much organic growth, we are fortunate, in the authors' opinions, to have a single, stable package at all. It is perhaps not surprising that documentation development has not kept pace. Yet with thousands of users [Gen], some (perhaps many) of whom are learning to program for the first time using NumPy, we assert that documentation is important. As the first author outlined at the SciPy 2008 Conference [Har], he found himself, in the fall of 2007, teaching data analysis with NumPy without much documentation, and resolved that, by the fall of 2008, his class would have the documentation it needed. He then founded the SciPy Documentation Project (SDP). The following sections describe the SDP, its current state, and our future needs and plans.

The Project

Before the start of the SDP, documentation entered NumPy and SciPy as did any component of the code: through the Subversion (SVN) repository. Only code developers could enter documentation. Other community members rarely made contributions, and if they did, it was by emailing the developers or a relevant mailing list. It was clear from the volume of unwritten documentation that a much larger group would need to be involved. We needed approved documentation writers, reviewers, and proofers (the SDP's three labor catagories) to enter, edit, discuss, pass judgement on, and proofread the documentation. We also needed tools for organizing the effort and generating progress statistics. Thus was born the SciPy Documentation Wiki [Vir], now located at docs.scipy.org.

After independently but concurrently commencing work on versions of the Wiki, Pauli Virtanen and Emanuelle Guillart quickly combined efforts, with Virtanen taking on the long-term management of the Wiki and the tools behind it. Gael Varoquaux and Stefan van der Walt also contributed. The Wiki interfaces with the SVN repositories, allowing semi-automatic commits into the NumPy and SciPy sources. Fully automatic commits are not permitted because of the risk of malicious code introduction. If code developers modify documentation in the sources, that gets communicated to the Wiki so that it has the current version.

The reference documentation appears as the Python docstring [Goo] to each documented object in the NumPy sources. These ASCII documents are written in a dialect of ReStructured Text [reST], which is quite readable without processing; the user sees the raw document in Python's help() facility. Prior to our taking the Wiki live, the community updated and refined the NumPy/SciPy Docstring Standard, to allow for formulae and figures. Because help() cannot display them, figures are discouraged in the NumPy documentation, but other packages adopting the template may permit them. The application Sphinx [Bran] produces HyperText Markup Language (HTML) and Portable Document Format (PDF) versions of the documentation, with indices. These processed forms are available at docs.scipy.org.

In addition to the public editing, commentary, and review system, there are instructions for participants on the front page, a recently-added Questions and Answers page, and a Milestones page that we use to organize and motivate the community, among other features. The system went live in the early Summer of 2008; since then, over 75 contributors have signed up for accounts. The Wiki generates weekly activity and participation statistics (http://docs.scipy.org/numpy/stats). Based on an examination of this page, we estimate that roughly 80% of the documentation has been written by about 13% of the contributors, but that most registered participants have contributed something.

Doc Coordinators

The SDP effort moves forward when the community works on it. To keep the community working, the first author has hired full-time Documentation Coordinators. Stefan van der Walt was the Coordinator in summer 2008, and continues his involvement in the project by handling documentation issues on the SciPy Steering Committee and working on issues related to SVN. The second author has been the Coordinator since June 2009. Among other duties, he has most recently been focused on updating the review protocol in anticipation of the NumPy doc review push (see below). As one can see from the Wiki's Status Page <http://docs.scipy.org/numpy/stats>`_, very little work was done between September 2008 and May 2009, during which period there was no Coordinator. (The fall 2009 hiatus is due to a delay in implementing a new dual-review protocol.)

The Coordinator keeps the community moving by defining milestones, defining short-term goals (like a weekly function category for everyone to work on), holding a weekly open teleconference with the writing community, and simply "talking it up" on the mailing lists to keep documentation on people's minds. The Coordinator also tends to write the pages nobody else wants to work on.

We also have motivators, in the form of T-shirts for those writing more than 1000 words or doing equivalent work such as reviewing, mention in NumPy's THANKS.txt file, and perhaps other incentives in the future. We are now soliciting contributions such as sponsored trips to the SciPy10 conference. Those interested in contributing sponsorships or ideas for incentives should contact the first author.

State of the Docs

The Wiki Status Page <http://docs.scipy.org/numpy/stats>`_ shows that there are over 2000 documentable functions, classes, and modules in NumPy, but that the sources contained just 8658 words of reference documentation when the SDP began, i.e., an average of a little more than four words per documentable object. Stefan van der Walt, the 2008 Documentation Coordinator, triaged the documentable objects into "Unimportant" and "Needs Editing" categories, and writing began.

Date	Needs Review	Being Edited	PDF Pages
Jun '08	6%	13%	<100
SciPy '08	24%	28%	309
SciPy '09	72%	6%	884

As of the SciPy '09 Conference (August 2009), the NumPy reference documentation consisted of over 110,000 words. Because the initial object triage was intentionally light, much of the final 15% of unedited objects are actually "Unimportant". This category mainly includes under-the-hood items that are never seen by normal users. With "submittable drafts" of the vast majority of reference pages, we will next focus on review.

Near Future: NumPy Review

The quality of the draft documentation is generally high, but writing is rarely complete without some critical review. The Project's original experiment with a one-review protocol resulted in very inconsistent documentation: some pages were approved by more technically-minded reviewers (such as developers), but had sections that were difficult to understand. Other approved pages, reviewed by professional writers, were missing important facts. We have thus defined and are implementing a two-review protocol, where both technical and presentation reviewers must approve each page. This requires changes to the Wiki (pending), changes to the reviewer instructions (finished), and recruitment of expert reviewers (we invite qualified readers to participate).

Next Projects

At the SciPy '09 Conference, the authors held a birds-of-a-feather (BoF) session where we discussed priorities for documentation after the NumPy and SciPy reference documentation is finished. The following ideas emerged:

NumPy User Guide: There are several pieces of text that a volunteer editor could integrate into a NumPy User Guide. The first are the concept pages in the numpy.doc module. These document NumPy concepts that occur in neither Python nor the most popular commercial scientific programming languages (e.g., Matlab or Interactive Data Language), such as broadcasting, indexing tricks, and other expediencies. There is a great deal of additional text in NumPy lead developer Travis Oliphant's original book, "Guide to NumPy." [Oli] Much of that work is technical documentation (C-API docs, etc.) or is appropriate for very expert users. This proposed User Guide would thus require much additional writing, but a great deal of excellent raw material already exists in the Wiki.

SciPyStack Test Drive (~10-pages): This document would narrate a demonstration that would give a first exposure to the power of NumPy. It would essentially be a marketing document, something one could give to colleagues to convince them it would be worth their time to learn more.

Getting Started Tutorial (~250 pages): This tutorial would teach a new user the basics of NumPy and some related packages, including array math, file I/O, basic plotting, some scientific methods packages, and where to learn more in all of these categories. The authors are aware of several such tutorials, but most are either unpublished or have been labeled by their authors as preliminary or unfinished. Others are outdated, having been written for one of NumPy's predecessor packages and not (yet) updated. Only a few are on the scipy.org web site. We may opt for one "General Tutorial" and several discipline-specific tutorials, such as the one for astronomy. [Gre]

Reading tools: The Python help() and NumPy np.info() functions are convenient, but are limited to ASCII text in the terminal. The numpy.info function should be enhanced so that, at the user's option, it can interface with external doc readers. For example, np.info(np.cos) might start a PDF reader and display the page that describes the np.cos function; or it might signal a Web browser to do the same using the HTML documentation.

SciPy Reference Pages: Documenting SciPy will be a monumental task. Though it starts from a much healthier state than did NumPy (at the time of article submission it had 628 pages of documentation, versus <100 for NumPy when the SDP began), the Doc Wiki shows that it has nearly twice as many documentable objects. Its content is highly technical, so writers with specialized knowledge will be required, arguably to a much higher degree than was needed for NumPy. Furthermore, much of the current documentation employs graduate-level technical language that many potential users may not understand. This work will begin in earnest once NumPy has been reviewed, but the SciPy portion of the Wiki is open, so those who are motivated, and in particular module maintainers, may begin this work at their convenience.

SciPyStack User Guide: This would be an integrative document to teach the fundamentals of the entire toolstack. We see it as covering basic (I)Python, giving pointers to introductory books on Python, teaching array basics (essentially the current NumPy User Guide content), presenting file I/O (including HDF, FITS, etc.), and having chapters on numerical computation, visualization, parallel computing, etc. To produce it, we propose following the academic "edited book" model. We would have a community discussion to agree on a table of contents and book conventions. A Request For Proposals would then solicit author teams to write chapters. An editor (or editors) would assemble and integrate the chapters, attending to things like style uniformity, and cross references, and indexing. We see as the final products a free e-book and a paper edition for retail sale. Our vision includes maintaining it to stay current.

What to Call It?

Readers will notice use of the term "SciPyStack" above. The community thinks of "SciPy" in two contexts: as a specific package and as the loosely defined collection (or stack) of packages used together for scientific computing in Python. The dual use causes confusion. For example, the authors have been involved in many discussions where someone strongly objected to a proposal, thinking that it was to add a major capability to the package when it was instead to produce a freestanding capability of which the unaltered package would be a component. As the community works on improving both the package and the stack, and as documentation moves to a stage where we are writing integrative material that must address both concepts together, it is our opinion that having separate words for the concepts will be beneficial. We offer here some thoughts and a proposal to initiate discussion.

Capitalization ("scipy" for the package vs. "SciPy" for the stack) has been insufficient, and few would wish to say "SciPy the stack" and "scipy the package" on each use. To avoid confusion, one of these usages much change. Changing the name of the package would shatter its API. We also have a web site, scipy.org, whose widely-known name we would be foolish to change, yet it is the portal to the stack. Our name for the stack would thus best begin with the fragment "SciPy..." for consistency with the well-known site.

We considered many possible names, including "Scientific Python" (already taken by another package) and "Py4Sci" (disagrees with web site name), and "SciPython" (likely to revert to "SciPy" in casual speech). "SciPyStack" could work; it is explicit, not easily abbreviated back to "SciPy", and short. Others might work as well. We raise the issue now to encourage discussion before writing begins on integrative documentation.

Conclusions

The SDP's first project has been reference documentation for NumPy. The vast majority of reference pages for user-visible objects are now ready for review, and these drafts have already been included in several NumPy releases. Given that most objects lacked documentation prior to the project, we feel these drafts are a significant improvement over the prior state, and the testimonials of students in courses taught by the first author bear out the notion that NumPy is much easier to learn today than it was two years ago. Once the NumPy documentation is finished, we will move on to SciPy. Future plans also include several books and a pamphlet. However, the big labor requirement now is for technical and presentation reviewers for the NumPy reference documentation.

There is another need, however: a sustainable funding stream to pay a Documentation Coordinator (which is all but essential for continued progress) and perhaps others in the future, such as book editors. The first author has been paying the Coordinators from his research grants, with the justification that this is ultimately costing the grants less than paying his group for the time they would require to learn NumPy without documentation. This cannot continue much longer, but the return to the community has already been significant. We believe that NumPy represents "bread and butter" to thousands of technical professionals worldwide, and that the number is quickly increasing. This software is not free of cost, only free of charge, and its continued development depends on contributions. We feel strongly that those who benefit substantially from NumPy, and especially those who profit from it, should consider contributing labor or funds. We are pursuing both commercial and governmental funding opportunities to continue supporting the SDP, and we seek senior collaborators for proposals. We invite those interested to contact us so that we may all continue to benefit from the power of the SciPy software stack.

Acknowledgements

Support for this work was provided by NASA through an award issued by JPL/Caltech.

References

[Bran] Georg Brandl, Sphinx: Python Documentation Generator. http://sphinx.pocoo.org/, 2009.

[Gen] Igor Genibel, Christoph Berg, Ian Lynagh (graphs), et al. Debian Quality Assurance: Popularity contest statistics for python-numpy. http://qa.debian.org/popcon.php?package=python-numpy, October, 2009.

[Goo] David Goodger and Guido van Rossum, Python Enhancement Proposal 257: Docstring Conventions. http://www.python.org/dev/peps/pep-0257/, 2001-9.

[Gre] Perry Greenfield and Robert Jedrzejewski, Using Python for Interactive Data Analysis. http://stsdas.stsci.edu/perry/pydatatut.pdf, 2007.

[Har] Joseph Harrington, The SciPy Documentation Project. In Proceedings of the 7th Python in Science Conference, G. Varoquaux, T. Vaught, and J. Millman (Eds.), pp. 33 – 35, http://conference.scipy.org/proceedings/SciPy2008/paper_7/full_text.pdf, 2008.

[Oli] Travis Oliphant, Guide to NumPy. http://www.tramy.us/numpybook.pdf, 2006.

[reST] reStructuredText: Markup Syntax and Parser Component of Docutils. http://docutils.sourceforge.net/rst.html, 2006.

[Vir] Pauli Virtanen, Emmanuelle Gouillart, Gael Varoquaux, and Stefan van der Walt, et al. pydocweb: A tool for collaboratively documenting Python modules via the web. http://code.google.com/p/pydocweb/, 2008-9.