Pandas

Goals

We propose funding the continued maintenance and development of Pandas (McKinney (2010)), a Python library providing high-performance, easy-to-use data structures.

Pandas is a foundational library in the Scientific Python Ecosystem, providing the most widely-used data structure for heterogenous, tabular data, and is the most-used Python tag on StackOverflow (Robinson (2019)). Pandas is widely used in industry and research, with over 1,000 citations (Mueller et al. (2019)).

Library Maintenance

Pandas is a large library with many users. We have many open issues (about 3,000) and Pull Requests (about 100). Keeping up with the stream of updates is time consuming, but important to the project. We would like to fund time dedicated specifically for maintenance. This would include

1. Promptly triaging newly opened issues
2. Promptly reviewing new pull requests
3. Ensuring open issue conversations are progressing
4. Ensuring that Pull Requests are not going stale and being abandoned

By dedicating time specifically to maintenance we hope to reduce the open issue backlog and increase the growth rate of new contributors, as pandas becomes an easier project to contribute to.

Native String Data Type

Currently, pandas stores text data in an `object`-dtype NumPy array. Each array stores Python strings. While pragmatic, since we rely on NumPy for storage and Python for string operations, this is memory inefficient and slow. We'd like to provide a native string type for pandas.

To solve the first issue, we propose a new extension type for string data. This will initially be opt-in, with users explicitly requesting `dtype="string"`. The array backing this string dtype may initially be the current implementation: an `object` -dtype NumPy array of Python strings.

To solve the second issue (performance), we'll explore alternative in-memory array libraries (for example, Apache Arrow). As part of the work, we may need to implement certain operations expected by pandas users (for example the algorithm used in, `Series.str.upper`). That work may be done outside of pandas (possibly within Apache Arrow).

Documentation Validation

To improve the quality and consistency of Pandas documentation, we've developed tooling to check docstrings in a variety of ways. Every docstring is checked for

1. Consistency: Ensuring that every docstring has the same components (parameters, return values, notes, examples, etc.). Ensuring that each section is formatted correctly.
2. Completeness: Ensuring that every function parameter is documented.
3. Correctness: Ensuring that the examples run correctly.

When a user submits a Pull Request to pandas, their changes to the documentation are automatically checked and informative error messages notify them of any issues.

Like many other projects, pandas uses the numpydoc standard for writing docstrings. With the collaboration of the numpydoc maintainers, we'd like to move the project-agnostic tooling we've written to a package outside of pandas, which any numpydoc-using project can benefit from.

If possible, we'd like this project to be undertaken by a member of an unrepresented minority, with mentorship provided by the pandas maintaiers. This project primarily requires experience with *using* pandas, NumPy, and related libraries, rather that deep knowledge of pandas' internals. Pandas has a diverse community, just not at the maintainer level (yet).

Performance Monitoring

Pandas uses airspeed velocity to monitor for performance regressions. ASV itself is a fabulous tool, but requires some additional work to be integrated into an open source project's workflow.

The asv-runner GitHub organization, currently made up of pandas maintainers, provides tools built on top of ASV. We have a physical machine for running a number of project's benchmarks and tools managing the benchmark runs and reporting on results.

We'd like to fund improvements and maintenance of these tools to

- Be more reliable. Currently, they're maintained on the nights and weekends when a maintainer has free time. This occasionally results in days or weeks of downtime.
- Tune the system to improve benchmark stability
- Report performance regressions to a project rather than relying on maintainers manually checking for regressions.
- Build a GitHub bot to request ASV runs *before* a Pull Request is merged. The benchmarks are too expensive to run as part of every commit. Running on-demand from a maintainer provides a nice balance.
- Pay for hosting a dedicated benchmark server. A maintainer's basement isn't the most stable environment for a machine.

This project has an impact beyond pandas. We run the benchmarks for many foundational libraries in the Scientific Python Ecosystem including xarray, Dask, scikit-image, scikit-learn, and PyMC3. Each of these projects would benefit from improvements made to the tooling.

Existing Support

Pandas currently has about two FTE.

- Anaconda Inc funds Tom Augspurger (50% FTE) and Brock Mendel (100% FTE)
- Ursa Labs funds Joris van den Bossche (50% FTE).
- NumFocus Small Development Grant for "Improving and modernizing the introductory *Getting Started* pages of the pandas documentation" ($5,000)

An up-to-date list of current members and institutional parters can be found in our governance documents.

References

McKinney, Wes. 2010. "Data Structures for Statistical Computing in Python." In *Proceedings of the 9th Python in Science Conference*, edited by Stéfan van der Walt and Jarrod Millman, 51–56.

Mueller, Andreas, Andy Terrel, Thomas Caswell, Ralf Gommers, Gina Helfrich, and Ryan Abernathey. 2019. "Mid-Scale Research Infrastructure - the Scientific Python Ecosystem," April. https://doi.org/10.6084/m9.figshare.8009441.v1.

Robinson, David. 2019. "Why Is Python Growing so Quickly?" *StackOverflow Blog*. StackOverflow. https://stackoverflow.blog/2017/09/14/python-growing-quickly/.