February 8, 2022

Draft version February 8, 2022
Typeset using LaTeX/show your work! **modern** style in AASTeX631

The Astropy Project: Sustaining and Growing a Community-oriented Open-source Project and the Latest Major Release (v5.0) of the Core Package

Astropy Collaboration

Submitted to ApJ

ABSTRACT

The Astropy Project supports and fosters the development of open-source and openly-developed `Python` packages that provide commonly-needed functionality to the astronomical community. A key element of the Astropy Project is the core package `astropy`, which serves as the foundation for more specialized projects and packages. In this article, we summarize key features in the core package as of the recent major release, version 5.0, and provide major updates for the project. We then discuss supporting a broader ecosystem of inter-operable packages, including connections with several astronomical observatories and missions. We also revisit the future outlook of the Astropy Project and the current status of Learn Astropy. We conclude by raising and discussing the current and future challenges facing the project.

Keywords: Astrophysics - Instrumentation and Methods for Astrophysics — methods: data analysis — methods: miscellaneous

1. INTRODUCTION

Author: *Adrian Price-Whelan*

The `Python` programming language is a high-level, interpreted (as opposed to compiled) programming language that has become an industry standard across many computational domains, technological sectors, and fields of research. Despite claims to the contrary (Portegies Zwart 2020), `Python` enables scalable, time- and energy-

Corresponding author: Astropy Coordination Committee
coordinators@astropy.org

The author list has two parts: the authors that made significant contributions to the writing of the paper in order of contribution, followed by contributors to the Astropy Project in alphabetical order. **The position in the author list does not correspond to contributions to the Astropy Project as a whole.** A more complete list of contributors to the core package can be found in the package repository, and at the Astropy team webpage.

efficient code execution (e.g., Augier et al. 2021) with a focus on code readability, ease of use, and interoperability with other languages. Over the last decade, `Python` has grown enormously in popularity to become a dominant programming language in the astronomical and broader scientific communities. For example, Figure 1 shows the number of yearly full-text mentions of `Python` as compared to a few other programming languages in refereed articles in the astronomical literature, demonstrating its nearly exponential growth in popularity. The rapid adoption of `Python` by astronomy researchers, students, observatories, and technical staff combined with an associated increase in awareness and interest about open-source software tools is contributing to a paradigm shift in the way research is done, data is analyzed, and results are shared in astronomy and beyond.

One of the factors that has led to its rapid ascent in popularity in scientific contexts has been the significant, volunteer-driven effort behind developing community-oriented open-source software tools and fostering communities of users and developers that have grown around these efforts. Today, a broad and feature-diverse "ecosystem" of packages exists in the `Python` scientific computing landscape: Roughly ordered from general-use to domain-specific, this landscape now includes packages that provide core numerical analysis functionality like `numpy` (Harris et al. 2020) and `scipy` (Jones et al. 2001–), visualization frameworks like `matplotlib` (Hunter 2007), machine learning and data analysis packages like `tensorflow` (Abadi et al. 2015), `pymc3` (Salvatier et al. 2016), and `emcee` (Foreman-Mackey et al. 2013), domain-specific libraries like `yt` (Turk et al. 2011), `plasmapy` (Community et al. 2021), `sunpy` (The SunPy Community et al. 2020), `Biopython` (Cock et al. 2009), and `sympy` (?) (to name a few in each category). The `astropy` (Astropy Collaboration et al. 2013, 2018) core package began in this vein, as an effort to consolidate the development of commonly-used functionality needed to perform astronomical research into a community-developed `Python` package.

The `astropy` core package was one of the first large, open-source `Python` packages developed for astronomy and provides, among other things, software functionality for reading and writing astronomy-specific data formats (e.g., FITS), transforming and representing astronomical coordinates, and representing and propagating physical units in code. An early description of the core functionality in `astropy` can be found in the first Astropy paper (Astropy Collaboration et al. 2013) or in detail in the core package documentation.[1] The `astropy` core package is now largely stable in that the software interface does not change without sufficient and significant motivation, and the addition of new features into the core package has slowed significantly as compared to the first years of its development. This is largely driven by the fact that the core package now represents just one piece of the broader astronomy `Python` context, and much new feature development is now happening in more specialized

[1] https://docs.astropy.org/

packages that are expanding the capabilities of the Astropy ecosystem by building on top of the foundations laid by the `astropy` core package. Because of this natural expansion, the name Astropy has grown in scope beyond a single `Python` library to become “the Astropy Project.”

The Astropy Project is a community effort that represents the union of the `astropy` core package, the ecosystem of astronomy-specific software tools that are interoperable with `astropy` (Astropy Affiliated Packages), *and* the community of users, developers, and maintainers that participate in Astropy efforts. However, there is no institution responsible for managing the Astropy Project, for funding or maintaining its development, or sustaining it into the future: The Project is maintained and coordinated largely by volunteers. While new Astropy-affiliated packages are being developed that expand upon the core functionality in the `astropy` package, representing a natural expansion of the Astropy Project ecosystem, the needs of and challenges faced by the Project are evolving. In particular, the transition from focusing our energy on development and maintenance of a single core package, to instead sustaining the core package and fostering the development of the community and its expansion has been a key issue faced by the Astropy Project in the last several years.

In this Article, we briefly describe recent key updates in the `astropy` core package since the last Astropy paper (“Paper II”; Astropy Collaboration et al. 2018), major updates in the governance, contributor base, and funding of the Project, and discuss some of the future plans and challenges faced by the Astropy Project.

2. MAJOR UPDATES TO THE ASTROPY CORE PACKAGE

2.1. *New Long-term Support (LTS) Version: v5.0*

Author: *Tom Robitaille*

Major versions of the core package – that is to say versions which add and/or modify functionality – are released approximately every six months, and are then maintained with releases that fix issues, until the next major version is released. However, every two years a major release is designated as a long-term support (LTS) release which continues to be maintained for up to two years (Tollerud 2013). The motivation for LTS releases is to provide longer-term stable versions of astropy that users requiring a high level of stability can make use of if they do not always need the latest features (this could include for example telescope pipelines and so on). The 5.0 release of the core package was designated as LTS, and since it was released at the end of 2021, it will be maintained until the end of 2023.

2.2. *Highlighted Feature Development*

Author: *Nathaniel Starkman, Marten van Kerkwijk*

Summarize major pieces of development since the last paper (since v2.0). A few ideas (taken from What’s New pages) below, but feel free to expand on or remove from this list:

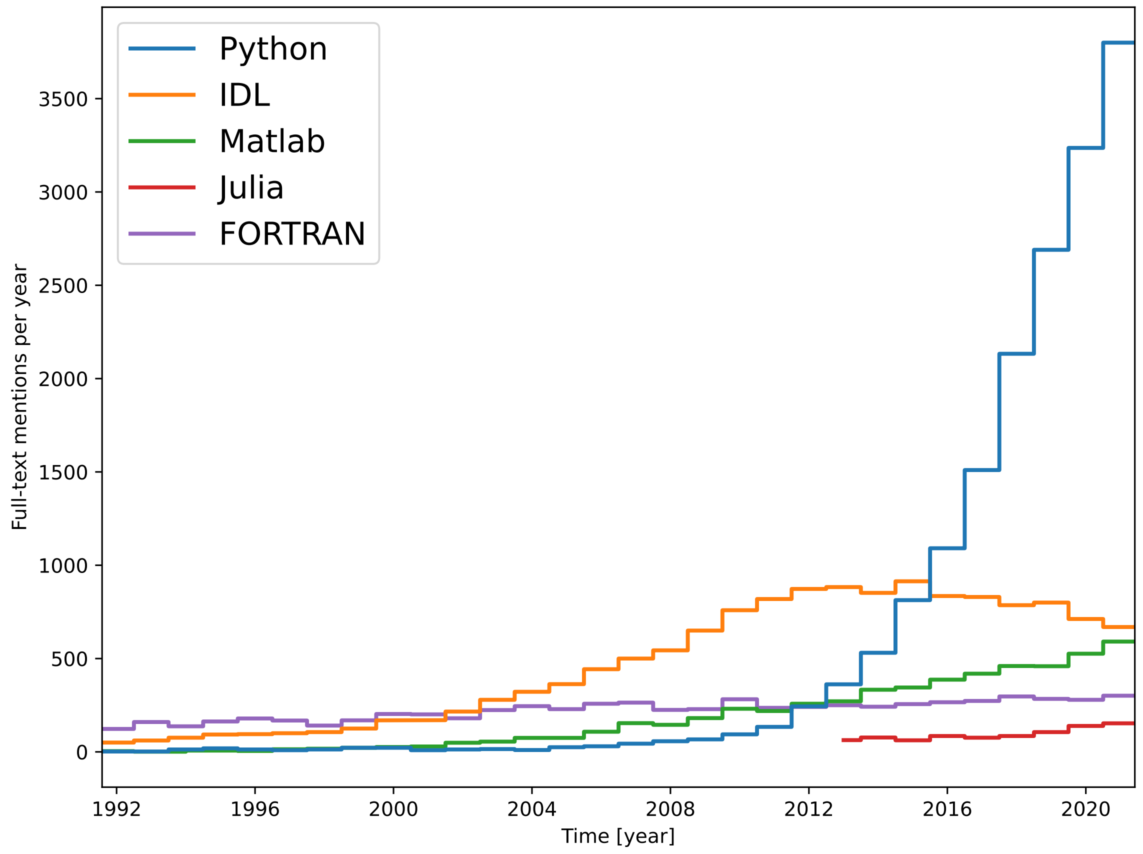

Figure 1. Yearly full-text mentions of programming languages (indicated in the figure legend) in refereed publications in the astronomical literature database in the Astrophysics Data System (ADS; **?**). `Python` has rapidly become the dominant programming language mentioned in refereed articles over the last 10 years.

- Support for representing and transforming velocity data in coordinates, epoch propagation (v3.0)
- Improved support for astronomical time series: TimeSeries object (v3.2), Box Least Squares periodogram (v3.1)
- Overall improved support of Quantity throughout numpy (v4.0) and scipy
- Native support for Time, Quantity, and SkyCoord objects in Astropy tables
- APE 14 WCS
- A new SpectralCoord class for representing and transforming spectral quantities
- Support for dask/large arrays in several places in the core package (4.1, 5.0, and maybe other versions)

3. MAJOR UPDATES IN THE ASTROPY PROJECT

3.1. *Project governance*

Author: *Erik Tollerud*

As part of the process of developing Astropy into a long-term sustainable product, and to improve transparency and accountability, the Project agreed to write down and formalize our governance structure (partly supported by explicit funding for this purpose - see §3.4). At the 2019 Astropy Coordination Meeting, input was gathered from participants on what governance structures existed in the associated Open Source Software communities, and what would fit well with the needs of Astropy. This led into a "retreat" planned for March 2020, but due to the COVID-19 pandemic, this became a series of virtual meetings of the "Astropy Governance Working Group". This group drafted the APE 0 document (Aldcroft 2021), which was then eventually ratified and implemented by the "Astropy Governance Implementation Working Group" in Fall 2021. While the process emphasized flexibility and the ability to adapt to changing circumstances, it is expected that this is the framework Astropy's governance will operate in for at least the medium-term future.

The APE0 (Aldcroft 2021) document lays out the principles of this governance structure, so we refer the reader to that document for a more thorough description. However here we highlight some key elements. While many of these principles were already de facto true or have been discussed organically (and have been discussed in earlier papers in this series), the APE0-based governance aimed to provide a single place where the community can agree as a starting point. With this in mind, it highlights the developer and user community of the Astropy Project as the ultimate sources of authority, as well as the core principle of "do-ocracy" that those who do work for the Project (be it coding, training, or other less concrete contributions) gain more influence on the outputs of the Project by virtue of their effort. APE0 adds, however, the concept of "voting members" - a self-governed part of that community who are entrusted to elect the Coordination Committee. While this committee has existed from the inception of the project, APE0 establishes a formal voting process for this committee, and explicitly outlines the rights and responsibilities of the Coordination Committee. This role is mainly to facilitate consensus and act as the decision maker when other mechanisms have failed. However, it also includes powers that either require central authority or secrets (e.g., passwords), but APE0 also charges the Committee to devolve responsibilities and seek community input on these items as often as possible.

The first Coordination Committee election under these rules took place in Fall 2021, electing a mix of prior existing and new coordination committee members, and was contested in the sense of more candidates than available slots. This suggests the process is already working to serve the long-term interests of the committee to both spread the coordination effort, and to ensure it is not dominated by the same people for as long as the Project continues. While other, more fine-grained governance improvements are planned for the future, it is clear the foundation is now in place.

3.2. *Contributor base*

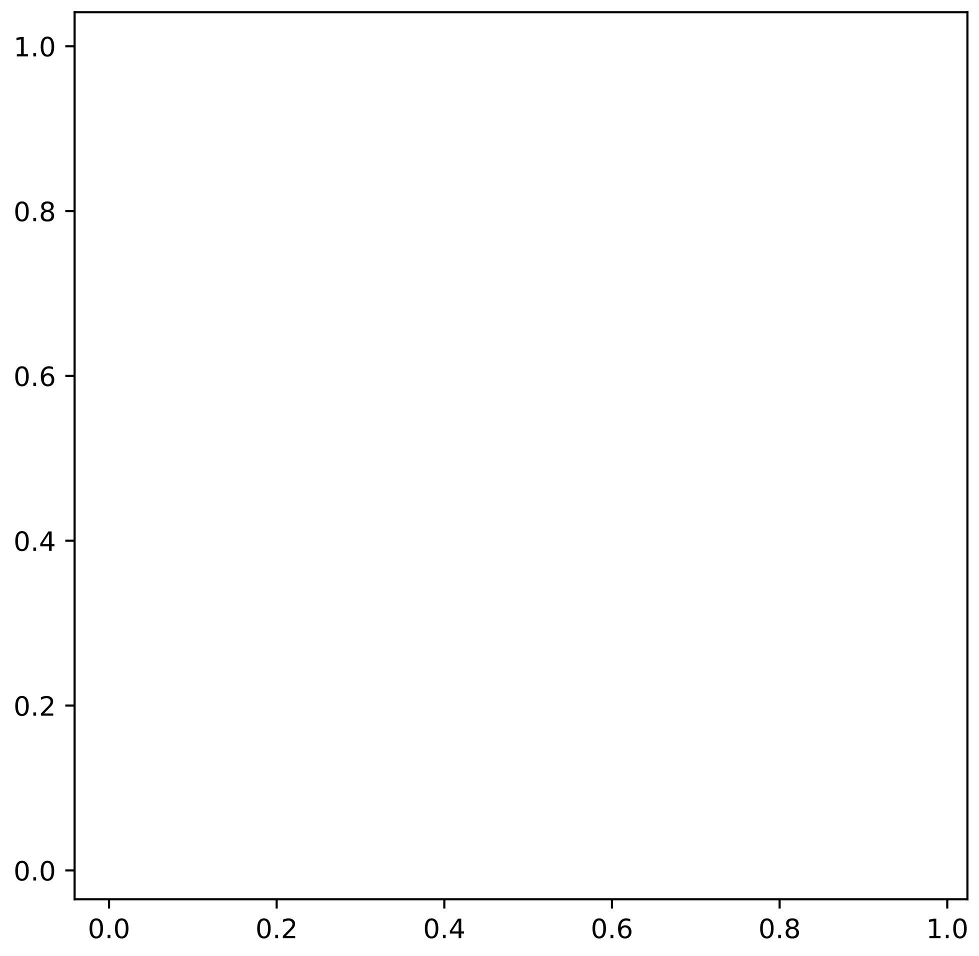

Figure 2. Placeholder figure!

Author: *Adrian Price-Whelan*

Overview and statistics of contributors. Highlight changes since v2.0.

3.3. *Inclusion, Diversity, and Equity Programs*

Author: *Lía Corrales*

3.4. *Funding*

Author: *Aarya Patil*

Summarize funding sources (Moore, NASA) and amounts and what this has been used for.

4. SUPPORTING THE ECOSYSTEM OF ASTRONOMICAL PYTHON SOFTWARE

4.1. *Community-oriented infrastructure*

Author: *Nicholas Earl*

The Astropy project supports the broader ecosystem by providing pre-configured infrastructure packages that the community can use to support and maintain their own software package infrastructure. These include tools to easily generate documentation and setup automated testing, as well as provide package scaffolding for new projects.

Sphinx is a common and useful tool for generating documentation for Python packages. The Astropy project maintains a default Sphinx configuration along with Astropy-specific extensions which can be easily added to community projects via the `sphinx-astropy` meta package. This tool provides a pre-configured Sphinx setup compatible with Astropy projects, which includes several extensions useful for generating API documentation (`sphinx-automodapi`), allowing for Numpy docstring parsing (`numpydoc`), embedded image handling (`sphinx-gallery`; `pillow`), advanced documentation testing support (`pytest-doctestplus`), and providing a custom documentation theme ideal for analysis packages (`astropy-sphinx-theme`).

Community package testing infrastructure is supported through the `pytest-astropy` meta-package, providing a unified testing framework with useful extensions compatible with both Astropy- and non-Astropy-affiliated community packages. This meta-package pulls in several `pytest` plugins to help with custom test headers (`pytest-astropy-header`), accessing remotely-hosted data files in tests (`pytest-remotedata`), interoperability with documentation (`pytest-doctestplus`), dangling file handle checking (`pytest-openfiles`), data array comparison support in tests (`pytest-arraydiff`), sub-package command-line testing support (`pytest-filter-subpackage`), improved mock object testing (`pytest-mock`), test coverage reports and measurements (`pytest-cov`), and configuring package for property-based testing (`hypothesis`).

The Astropy Package Template helps facilitate the setup and creation of new Python packages leveraging the Astropy ecosystem. This tool utilizes the Cookiecutter project to walk users through the process of creating new packages complete with documentation and testing support. Additionally, the package template generation process includes the ability to setup interoperability with GitHub, allowing for easy repository access from documentation, as well as an example GitHub Actions workflow to demonstrate the use of GitHub's continuous integration tooling.

4.2. *Affiliated packages*

Author: *Matt Craig, Brett Morris*

Highlight a few new affiliated packages and major updates to existing ones. Include a big table of all affiliated packages and references (as in v2.0 paper).

4.3. *Connections with data archives*

Author: *Adam Ginsburg*

Summary of Astroquery and engagement with archives. Cite astroquery paper.

4.4. *Connections with Observatories and Missions*

Brief summary of efforts in observatory- or mission-driven development that have contributed to Astropy, and vice versa.
JWST Author: *Larry Bradley*
Gemini Author: *Looking for volunteers!*
Cherenkov Telescope Array Author: *Axel Donath, Max Noethe*
Rubin Observatory Author: *Looking for volunteers!*
LIGO/Virgo/KAGRA Author: *Leo Singer*

5. FUTURE PLANS FOR THE ASTROPY PROJECT

5.1. *Roadmap*

Author: *Clara Brasseur*
Overview of roadmap and some highlights.

6. LEARN ASTROPY

Author: *Lía Corrales, David Shupe + Learn team*

- Current status and scope
- Vision for the future (review what we said in v2.0 paper)
- User forums and engagement with user base
- Workshops

6.1. *Current and Future Challenges*

Author: *Looking for volunteers!*

- Attracting new contributors when the code has become quite complex,
- Contributor to maintainer mentoring,
- Long-term / sustained funding for maintaining infrastructure,
- ...

We would like to thank the members of the community that have contributed to Astropy, that have opened issues and provided feedback, and have supported the project in a number of different ways.

The Astropy community is supported by and makes use of a number of organizations and services outside the traditional academic community. We thank Google for financing and organizing the Google Summer of Code (GSoC) program, that has funded severals students per year to work on Astropy related projects over the summer. These students often turn into long-term contributors. We also thank NumFOCUS and the Python Software Foundation for financial support. Within the academic community, we thank institutions that make it possible that astronomers and other developers on their staff can contribute their time to the development of Astropy projects. We would like acknowledge the support of the Space Telescope Science Institute, Harvard–Smithsonian Center for Astrophysics, and the South African Astronomical Observatory.

Furthermore, the Astropy packages would not exist in their current form without a number of web services for code hosting, continuous integration, and documentation; in particular, Astropy heavily relies on GitHub, Travis CI, Appveyor, CircleCI, and Read the Docs.

`astropy` interfaces with the SIMBAD database, operated at CDS, Strasbourg, France. It also makes use of the ERFA library (Tollerud et al. 2017), which in turn derives from the IAU SOFA Collection[2] developed by the International Astronomical Union Standards of Fundamental Astronomy (Hohenkerk 2011).

Software: `astropy` (Astropy Collaboration et al. 2013, 2018), `numpy` (Harris et al. 2020), `scipy` (Jones et al. 2001–), `matplotlib` (Hunter 2007), `Cython` (Behnel et al. 2011).

REFERENCES

Abadi, M., Agarwal, A., Barham, P., et al. 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/

Aldcroft, Tom, C. M. C. K. C. L. C. S. F. N. L. P. L. M. S. P.-W. A. R. T. S. D. S. B. . T. E. 2021, Astropy Proposal for Enhancement 0: The Astropy Project Governance Charter (APE 0), doi: 10.5281/zenodo.4552791

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33, doi: 10.1051/0004-6361/201322068

Astropy Collaboration, Price-Whelan, A. M., Sipőcz, B. M., et al. 2018, AJ, 156, 123, doi: 10.3847/1538-3881/aabc4f

Augier, P., Bolz-Tereick, C. F., Guelton, S., & Mohanan, A. V. 2021, Nature Astronomy, 5, 334, doi: 10.1038/s41550-021-01342-y

Behnel, S., Bradshaw, R., Citro, C., et al. 2011, Computing in Science Engineering, 13, 31 , doi: 10.1109/MCSE.2010.118

Cock, P. J. A., Antao, T., Chang, J. T., et al. 2009, Bioinformatics, 25, 1422, doi: 10.1093/bioinformatics/btp163

Community, P., Everson, E., Staczak, D., et al. 2021, PlasmaPy, 0.6.0, Zenodo, doi: 10.5281/zenodo.4602818

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. 2013, PASP, 125, 306, doi: 10.1086/670067

Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Nature, 585, 357, doi: 10.1038/s41586-020-2649-2

Hohenkerk, C. 2011, Scholarpedia, 6, 11404, doi: 10.4249/scholarpedia.11404

Hunter, J. D. 2007, Computing In Science & Engineering, 9, 90, doi: 10.1109/MCSE.2007.55

Jones, E., Oliphant, T., Peterson, P., et al. 2001–, SciPy: Open source scientific tools for Python. http://www.scipy.org/

Portegies Zwart, S. 2020, Nature Astronomy, 4, 819, doi: 10.1038/s41550-020-1208-y

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. 2016, PeerJ Computer Science, 2, e55, doi: 10.7717/peerj-cs.55

The SunPy Community, Barnes, W. T., Bobra, M. G., et al. 2020, The Astrophysical Journal, 890, 68, doi: 10.3847/1538-4357/ab4f7a

Tollerud, E. 2013, Astropy Proposal for Enhancement 2: Astropy Release Cycle and Version Numbering (APE 2), doi: 10.5281/zenodo.1043888

Tollerud, E., Pascual, S., Nair, P., et al. 2017, doi: 10.5281/zenodo.1021149

Turk, M. J., Smith, B. D., Oishi, J. S., et al. 2011, ApJS, 192, 9, doi: 10.1088/0067-0049/192/1/9