Astropy: Powering Astronomy with Python

Name: Arpit Agarwal

Date: 05/10/2025

1. Astropy is an open-source core Python package developed by and for the community with a purpose to serve astronomy and astrophysics.It is primarily concerned with providing standardized, efficient, and robust tools to professional astronomers and researchers.It has a wide range of functionalities including unit conversion, celestial coordinate systems, time conversion, FITS file operations, cosmology calculations, etc. Astropy offers a library of platforms to build advanced pipelines of observational and theoretical astrophysics through the integration of many tasks individually completed by a mixture of ad hoc software and homegrown scripts.
2. I selected Astropy because I was particularly interested in its model and fit system, which gives a convenient mechanism for specifying analytical models and fitting them to data. This is essential throughout most of astronomy, such as spectral line fitting, photometric curve fitting, and kinematic modeling. The systematic approach to specifying models, combining them, and using optimization routines made it both powerful and convenient. Along with its common use by the community of astronomers, this facet served to distinguish Astropy to me as a utility and learning tool that was worth exploring.
3. Astropy originally came out in 2011 as a mother project to bring together numerous disjoint astronomy-specific Python packages into a single, properly documented library. Astropy now has developed into the foundation of the Python astronomy stack. Before Astropy, astronomers used to utilize single packages or IDL routines for FITS file manipulations and coordinate conversions. For general modeling and fitting, SciPy—more precisely scipy.optimize.curve_fit—has been the standard for generic data fitting. However, Astropy's modeling subpackage is a more organized and astronomy-focused solution, with features such as integrated analytical models, unit support, and the ability to compose models. Astropy has also influenced or integrated similar packages like photutils and specutils, extending its functionality through a larger ecosystem.
4. Yes, Astropy is actively maintained by a large collaborative community under the Astropy Project, not one author. While a few of the original authors remain active, the project has grown with hundreds of contributors worldwide. Its development takes place on GitHub, where users can readily obtain information on how to contribute, including reporting bugs, making pull requests, writing tests, and improving documentation. The project also has regular releases and formal governance, indicating a high level of long-term maintenance and community involvement.

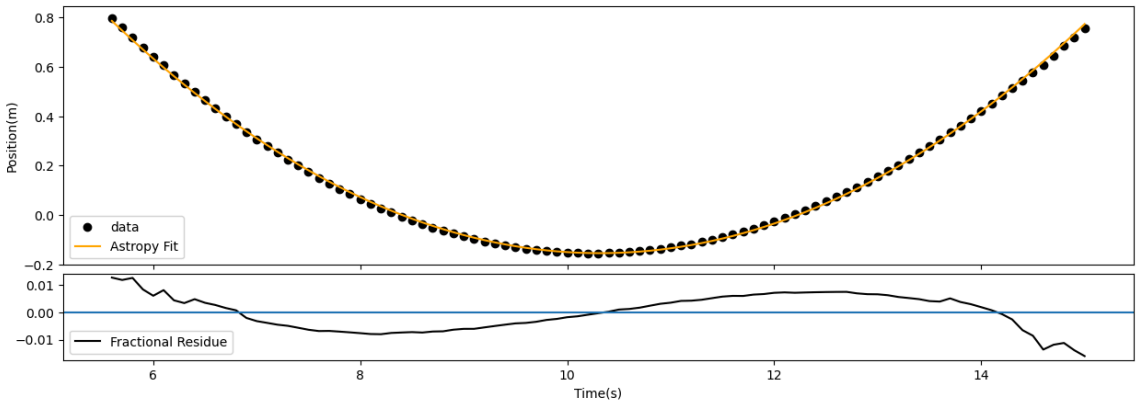
5. Installing Astropy was smooth and well-documented. I followed the official recommendation from the Astropy website and used the command below, which installs the core package along with optional but useful dependencies. The installation completed without any problems, and the package was ready to use immediately.

!pip install astropy[recommended] --upgrade

6. As explained above it could be directly installed with a pip/conda command
7. Yes, the source code for Astropy is fully open-source and publicly available. While running pip install astropy[recommended] installs the package for use, the actual codebase can be inspected and contributed to via its GitHub repository at https://github.com/astropy/astropy. The repository is well-organized, with clearly labeled subdirectories for each module (e.g., modeling, io, units), and includes documentation, issue tracking, and contribution guidelines. This transparency makes it easy to understand the internal workings of the package, troubleshoot, or even extend it for custom use.
8. Yes, Astropy is used as a base library by many other Python astronomy packages. A good example is Photutils, which uses Astropy's data structures and units to perform aperture and point-spread function (PSF) photometry. Another is Specutils, which is a package for spectroscopic data analysis that relies on Astropy's SpectralCoord, units, and modeling tools. Many other packages in the broader Astropy ecosystem and standalone astronomy projects depend on Astropy due to its stability and community conventions. Its popularity also makes it a center of modern astronomical software development.
9. Astropy is primarily designed to be called within Python scripts and Jupyter notebooks, making it well-suited for interactive data analysis and reproducible research. It is not well-suited for command-line or web-based interfaces, though it can be part of larger applications with those front-ends. The package's tool support for packages like NumPy, SciPy, and Matplotlib, together with its module-based design, also render it well adapted for Jupyter settings, where result, code, and visualisation can be amalgamated into a single environment. The bulk of official tutorial and example descriptions are notebooks by nature, driving this application as well. Below is one example of the custom model creation from astropy.modelling

@custom_model def s_astropy(t,s0=0.0,v0=0.0,a0=0.0): return s0+v0*t+(1/2)*a0*(t**2)

10. In the accompanying Jupyter file I have used the Model and Fitting line of codes from the Astropy Module
11. The package doesn't produce images on its own but can be easily used with numpy and matplotlib to create figures as used in the code.
12. Below is a figure of the fit done by Astropy for the data collected while dropping an object from an inclined planed using an airtrack compared with the original data:



13. Astropy is primarily written in Python but includes some C extensions for performance optimization, which are automatically handled during installation.
14. The input to Astropy typically consists of parameters (e.g., model attributes, units, coordinates) and datasets like FITS files, but it can also generate synthetic data for modeling and testing.
15. The output from Astropy can include fitted models, transformed coordinates, unit-aware quantities, tables, or data written to files (like FITS), as well as screen outputs for summaries and logs.
16. Yes, Astropy includes a comprehensive suite of unit tests, regression tests, and continuous integration (CI) checks to ensure code reliability and stability across updates.
17. When installed, Astropy suggests testing functionality by running its internal test suite using python -c "import astropy; astropy.test()", a standardized set of unit tests on submodules. Tests check for correctness, stability, and compliance with expected behavior. Performance confidence is also handled by developers and contributors using benchmarking tools like asv (Airspeed Velocity).
18. Astropy depends on several core Python packages such as NumPy, SciPy, and Matplotlib for numerical operations, optimization, and visualization; I identified these dependencies through the official documentation and by inspecting the install_requires section in the setup.cfg file on GitHub.
19. Astropy provides extensive and well-organized documentation, including user guides, API references, tutorials, and example Jupyter notebooks, which I found more than sufficient for learning and applying its tools effectively.
20. Astropy gives the following references for citations:

- \software{Astropy \citep{astropy:2013, astropy:2018, astropy:2022}} (For AAS Journals)
- This work made use of Astropy:\footnote{http://www.astropy.org} a community-developed core Python package and an ecosystem of tools and resources for astronomy \citep{astropy:2013, astropy:2018, astropy:2022}. (For other Journals)
- Citation can also be added witht he code: astropy.**citation**

22. The papers that refered Astropy were:

- "The Gaia Early Data Release 3: Summary of the contents and survey properties"** (Gaia Collaboration, 2021) uses Astropy for data analysis and coordinate transformations.
- "Astroquery: An Astronomical Web-querying Package in Python"** (Ginsburg et al., 2019) cites Astropy as a core dependency and foundation for handling data structures and units.

23. There was nothing new I learnt in this module except models and model making. The course was mostly sufficent
24. I was using this package for the first time and wasn't part of any group