Astropy.Coordinates: Mapping a Globular Cluster System

By: Anthony Angarita

(1) This package was the .Coordinates package within Astropy. Its goal is to aid with the astrophysical functions within Python to make it easier. It does this a few ways by defining astronomical positions, converting between astronomical coordinate systems, and accessing 3d spatial information.

(2) I selected this package because when I was looking through the list that we were provided with packages to use I saw astropy and remembered that we used it early in the semester. I figured that because we had used it before, my operating system must be able to handle it. Then in looking into what could be done with astropy I found that making a 3D plot of astrophysical bodies was possible and in looking into how to do so found astropy.coordinates. One might say that I did the process a little backwards by finding something that I wanted to do and then narrowing it down to the astropy package that best suited my needs.

(3) Astropy is 12 years old and was initially released publicly in March of 2013. Astropy.coordinates was a part of the core release. The version of astropy that I installed was 7.0.1. I found that before astropy in order to complete the project that I did, a coder might use astrolib.coords which is a part of an old library for astronomical analysis called stsci_python.

(4) Astropy is still maintained and is currently being worked on as a community project. The original authors, Erik Tollerud and Tom Robitaille, are still listed under many of the roles on the project but overall it has become a community effort. In order to contribute to the project you must pick an issue that can be found on the astropy website and create your fix for it in a forked dev environment in GitHub. Then you must submit a pull request with tests for your fix and documentation that details what you updated or changed about the code. Then your submission is checked by someone else higher up in the astropy community.

(5) Astropy was very easy to install and use.

(6) I installed it using the standard “!pip install” function and got what I needed from the coordinates packages by using "from and "import ".

(7) The source code is available on GitHub at https://github.com/astropy/astropy

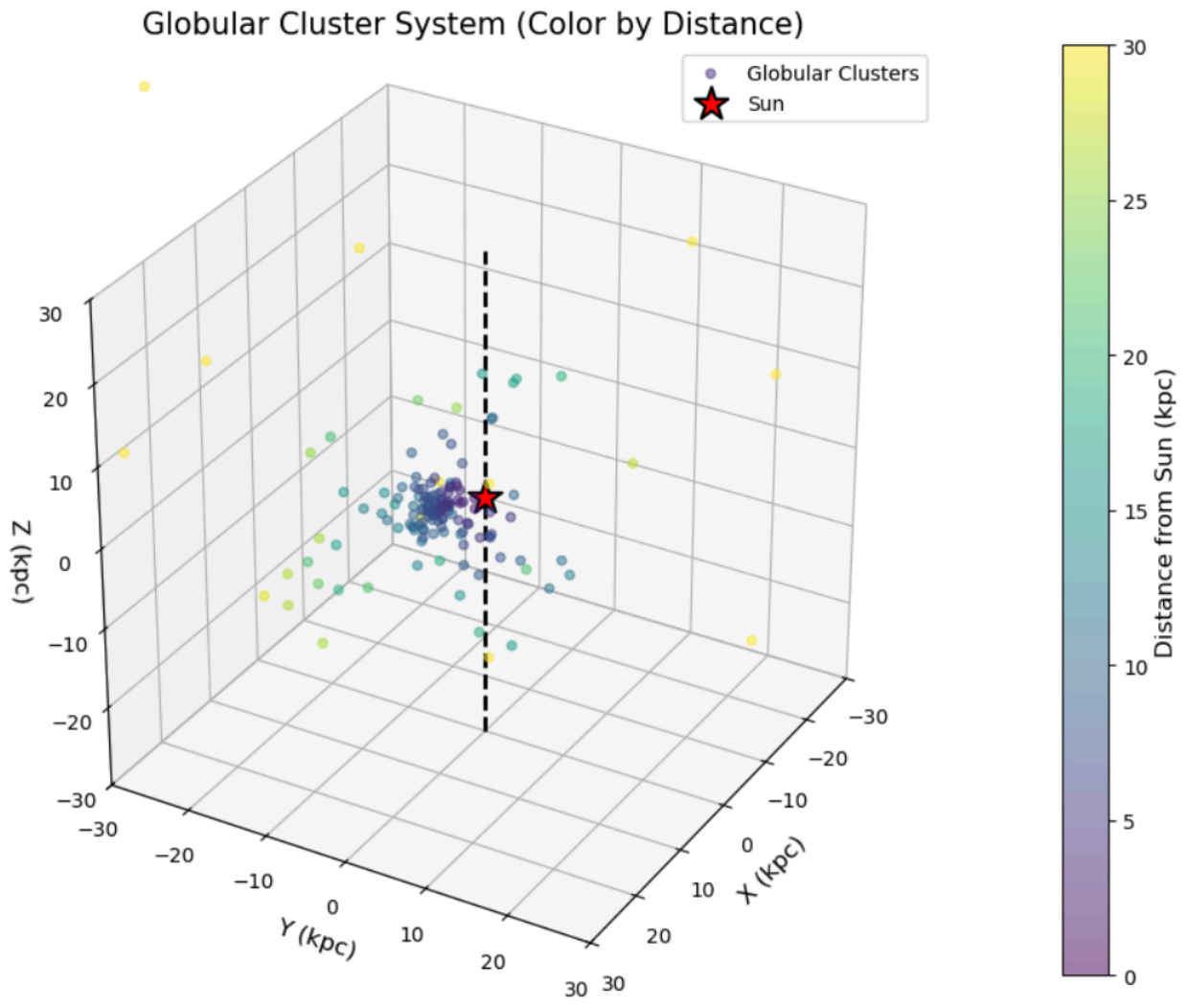
(8) The code is used by other packages. Astroquery and sunpy both use it..

(9) This code is used in Python scripts and Jupyter notebooks but does not have any command-line or web interface specifically for the astropy.coordinates package.

(10) Below is a line of code that saves the Right Ascension values and Declination values using units that are specified. Then it saves the distance values and defines that it is measured in the International Celestial Reference System (icrs). It saves all of this info under “coords”.

```
coords = astropy.coordinates.SkyCoord(ra=df['RA '].values, dec=df['DEC'].values, unit=(u.hourangle, u.deg), distance=df['R_Sun'].astype(float) *u.kpc, frame = 'icrs')
```

(11) Astropy itself does not create plot but works seamlessly with matplotlib.



(12) This figure is a 3D plot of globular clusters centered around the Sun with the Sun being marked with a red star and an axis going through the center of it. The axes are in units of kiloparsecs (kpc) and the data was pulled from the Harris Catalog of Galactic Globular Clusters.

(13) Astropy is not entirely Python as it does use code from C for a few astronomical routines.

(14) The input parameters that I use for these packages are Right Ascension, Declination, distance, frame, and time. It also takes observer information like Earth location and can use arrays for catalogs.

(15) This tool outputs SkyCoord objects, cartesian coordinates, angular separations, and transformed frames.

(16) The code does provide unit tests. The Astropy suite has a lot of unit tests especially for coordinates. The continued work on the project ensures compatibility.

(17) I can feel confident about this package producing a reliable result for a few reasons. It is being constantly checked and worked on, it is validated against external catalogs and tests, and it is trusted and cited by professional astronomers.

(18) This code depends on numpy, scipy, astropy.units, and erfa. I found this out initially by using (coord.__dependencies__) and then also found it by checking "requirements.txt." on GitHub)

(19) The documentation that is provided is more than enough. It has documents and more importantly web pages with walkthroughs of basic uses. These I found to be more than enough.

(20) Astropy requests citations done the following way: "Astropy Collaboration et al. 2013, A&A,"558, A33 (ADS)"

(21) In my notebook, I use pandas to read in my data, matplotlib.pyplot to plot my data, and astropy.units to do some unit conversions.

(22) I found two papers that use this package. **Price-Whelan et al. (2018), Gala: A Python package for galactic dynamics** and **Green et al. (2019), A 3D Dust Map Based on Gaia and Pan-STARRS**

(23) I did not have to learn any new methods for this project. I made use of pandas and matplotlib as we had used them in class and in our labs. All I needed to learn really was what the inputs and outputs of the new tools I was using.

(24) I had a small amount of prior experience using astropy before this because we did a few minor demonstrations with it during lectures in class. I did not work within a group.