Astronomical Data in Python

Astronomical Data in Python is an introduction to tools and practices for working with astronomical data. Topics covered include:

- Writing queries that select and download data from a database.
- Using data stored in an Astropy `Table` or Pandas `DataFrame`.
- Working with coordinates and other quantities with units.
- Storing data in various formats.
- Performing database join operations that combine data from multiple tables.
- Visualizing data and preparing publication-quality figures.

As a running example, we will replicate part of the analysis in a recent paper, "[Off the beaten path: Gaia reveals GD-1 stars outside of the main stream](#)" by Adrian M. Price-Whelan and Ana Bonaca.

This material was developed in collaboration with [The Carpentries](#) and the Astronomy Curriculum Development Committee, and supported by funding from the American Institute of Physics through the American Astronomical Society.

I am grateful for contributions from the members of the committee – Azalee Bostroem, Rodolfo Montez, and Phil Rosenfield – and from Erin Becker, Brett Morris and Adrian Price-Whelan.

The original format of this material is a series of Jupyter notebooks. Using the links below, you can read the notebooks on NBViewer or run them on Colab. If you want to run the notebooks in your own environment, you can download them from this repository and follow the instructions below to set up your environment.

Prerequisites

This material should be accessible to people familiar with basic Python, but not necessarily the libraries we will use, like Astropy or Pandas. If you are familiar with Python lists and dictionaries, and you know how to write a function that takes parameters and returns a value, that should be enough.

We assume that you are familiar with astronomy at the undergraduate level, but we will not assume specialized knowledge of the datasets or analysis methods we'll use.

Notebook 1

This notebook demonstrates the following steps:

1. Making a connection to the Gaia server,
2. Exploring information about the database and the tables it contains,
3. Writing a query and sending it to the server, and finally
4. Downloading the response from the server as an Astropy `Table`.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 2

This notebook starts with an example that does a "cone search"; that is, it selects stars that appear in a circular region of the sky.

Then, to select stars in the vicinity of GD-1, we:

- Use `Quantity` objects to represent measurements with units.
- Use the `Gala` library to convert coordinates from one frame to another.
- Use the ADQL keywords `POLYGON`, `CONTAINS`, and `POINT` to select stars that fall within a polygonal region.
- Submit a query and download the results.
- Store the results in a FITS file.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 3

Here are the steps in this notebook:

1. We'll read back the results from the previous notebook, which we saved in a FITS file.
2. Then we'll transform the coordinates and proper motion data from ICRS back to the coordinate frame of GD-1.
3. We'll put those results into a Pandas `DataFrame`, which we'll use to select stars near the centerline of GD-1.
4. Plotting the proper motion of those stars, we'll identify a region of proper motion for stars that are likely to be in GD-1.
5. Finally, we'll select and plot the stars whose proper motion is in that region.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 4

Here are the steps in this notebook:

1. Using data from the previous notebook, we'll identify the values of proper motion for stars likely to be in GD-1.
2. Then we'll compose an ADQL query that selects stars based on proper motion, so we can download only the data we need.
3. We'll also see how to write the results to a CSV file.

That will make it possible to search a bigger region of the sky in a single query.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 5

Here are the steps in this notebook:

1. We'll reload the candidate stars we identified in the previous notebook.
2. Then we'll run a query on the Gaia server that uploads the table of candidates and uses a `JOIN` operation to select photometry data for the candidate stars.
3. We'll write the results to a file for use in the next notebook.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 6

Here are the steps in this notebook:

1. We'll reload the data from the previous notebook and make a color-magnitude diagram.
2. Then we'll specify a polygon in the diagram that contains stars with the photometry we expect.
3. Then we'll merge the photometry data with the list of candidate stars, storing the result in a Pandas `DataFrame`.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Notebook 7

Here are the steps in this notebook:

1. Starting with the figure from the previous notebook, we'll add annotations to present the results more clearly.
2. The we'll see several ways to customize figures to make them more appealing and effective.
3. Finally, we'll see how to make a figure with multiple panels or subplots.

Press this button to run this notebook on Colab:

[or click here to read it on NBViewer](#)

Installation instructions

Coming soon.