Suyog Garg
suyog9999sg@gmail.com
+91 7358246475

Add a CDS-ASCII writer to astropy

Google Summer of Code (GSoC) 2021 | OpenAstronomy – astropy | Mentors: Aarya Patil and Hans Moritz Günther

12 Apr 2021

Introduction

I completed my BTech in Mechanical Engineering from IIITDM Kancheepuram, Chennai, in October last year. I have a keen interest in Astrophysics and thus, I have previously worked on a few theoretical, observational and computer intensive projects relating to the field. Alongside my Masters studies, I am also concurrently working on two research projects. My resume and the Personal Information section below describes these in more detail along with shedding light on my previous research experiences, skills and abilities. My motivation in applying for Google Summer of Code (GSoC) through this Astropy project is that my academic interests, combined with my skills, enable me to provide some useful contributions towards the Astropy library. As I am well aware through personal experience, such feature implementations are routinely used by numerous scientists and researchers worldwide.

The projects I am doing presently would be finished soon and I would be able to put-in around 40 hours of work a week for the GSoC project. I primarily use a Linux machine with 8 GB of RAM and an Intel core-i7 7th gen processor for my work. I am also available for the whole duration of the GSoC project and have access to continuous reliable internet connectivity.

This proposal starts with a Personal Information section giving an overview of my academic background in the next section. It also tries to highlight how I am an apt candidate for the aforementioned project. This is followed by a brief synopsis of the intended project. In Section 4, the CDS-ASCII data format and Astropy writing mechanisms are detailed, together with how a CDS-ASCII writer can be implemented in Astropy. My prior contributions to Astropy relating to the topic hand are also discussed. Section 5 talks of the tentative timeline the project can follow. Finally the proposal ends with some concluding notes and a references section.

Personal Information

Overview

Name	Suyog Garg
Email	suyog999sg@gmail.com
Address	Mandla, Madhya Pradesh 481661 India
Mobile	+91 7358246475
Education	
Institute	
GitHub	github.com/Suyog7130
LinkedIn	linkedin.com/in/suyog-garg-9443b611b/
NASA ADS	NASA ADS link
Google Scholar	scholar.google.com/suyog-garg
Blog	suyog20.blogspot.com/
Resume	

Skills

Programming Languages	C, Python, MATLAB/Octave, Bash, Git, HTML, Julia, Familiar with FORTRAN
Specific Python Libraries	numpy, scipy, matplotlib, pandas, astropy, sunpy, tensorflow, keras
Softwares and Tools	MATLAB Simulink, LabVIEW, Multisim, LaTeX, Catia, Ansys, Adobe Photoshop
Spoken Languages	Hindi, English, Urdu, Japanese, German, Tamil

Awards and Honours

- **Nominated for the award of Embassy-recommended MEXT Scholarship 2021**, *Embassy of Japan in India*, **October 2020.**
- **Long-term Project Studentship**, *Tata Institute of Fundamental Research (TIFR), Mumbai*, India, **June 2020.**
- **Vacations Student Program (VSP),** *Inter-University Center for Astronomy and Astrophysics (IUCAA), Pune*, **May 2020.**
- **Physics Summer Research Intern,** *Institute of Mathematical Sciences (IMSc), Chennai*, **May 2019.**

Research Experience

I am currently working as a remote project student with the Helioseismology group at Tata Institute of Fundamental Research (TIFR), Mumbai. My focus here is on studying Sunspots Magnetograms. This project, along with my past research works, heavily rely on computational analysis, often utilizing capabilities of software libraries and pipelines like Astropy.

List of publications

1. S. Bapat et al. **EinsteinPy: A Community Python Package for General Relativity.** *arXiv e-prints*, May 2020, 2005.11288. URL https://arxiv.org/abs/2005.11288
2. S. Garg, B. B. Karak, R. Egeland, W. Soon, and S. Baliunas. **Waldmeier Effect in Stellar Cycles.** *ApJ*, 886(2):132, Dec 2019, 1909.12148. URL https://doi.org/10.3847/1538-4357/ab4a17
3. S. Garg and M. Bagchi. **A Negligible Tidal Effect in a Periastron Precession in Neutron Star–Black Hole (Stellar Mass) Binaries.** *Research Notes of the American Astronomical Society*, 3(9):125, Sep 2019 URL https://doi.org/10.3847/2515-5172/ab3faf

Project Synopsis

Title	Add a CDS-ASCII writer to astropy
Website	openastronomy.org/gsoc/gsoc2021/
Mentors	Aarya Patil and Hans Moritz Günther

Astropy is a massive Python library providing methods and general purpose tools for processing and analysis of data generated in astronomy and related fields. It is widely used by researchers and the scientific community at large. One important feature of Astropy is reading and writing tabular data in a wide variety of useful formats. One such astronomical data storage format is the CDS-ASCII format employed by Centre de Données astronomiques de Strasbourg (CDS) for maintaining its VizieR catalog. Currently Astropy only supports reading data from a CDS-ASCII table and not writing to it. The present project is meant to address this issue by adding a CDS-ASCII writer to Astropy. CDS already has an available package to generate standardized ASCII tables for submission of data to its catalog. The proposed Astropy CDS-ASCII writer can be based on the methods used by that package.

Project Description

Main Aim

As mentioned in the Project Synopsis above, as of now Astropy only has a method to read a CDS-ASCII format data table. The main aim of this project is to add a feature to Astropy for writing to a CDS-ASCII format table. A secondary aim of the project would be to check for and correct any bugs in the implementation and to make proper documentation for the newly added methods.

Details

The CDS-ASCII format

Established in 1972, Centre de Données astronomiques de Strasbourg or CDS for short, is a data center located in France that collects astronomical data. [3] It provides some of the world's most widely used astronomical data cataloging services in the form of its VizieR and SIMBAD database services. CDS' databases use an eponymous ASCII file format for storage named CDS-ASCII format, instead of the usual Flexible Image Transport Format (FITS) used throughout the astronomy field. A FITS format data storage is not feasible for large tabular data like the CDS catalogs because,

- CDS stores data *published* in astronomical publications and it is required that the files extracted be comparable to the tables in the publication.
- FITS description of the table structure is fine tuned for computer usage only.
- FITS tables are of larger size than ASCII files due to fixed-length record structure.

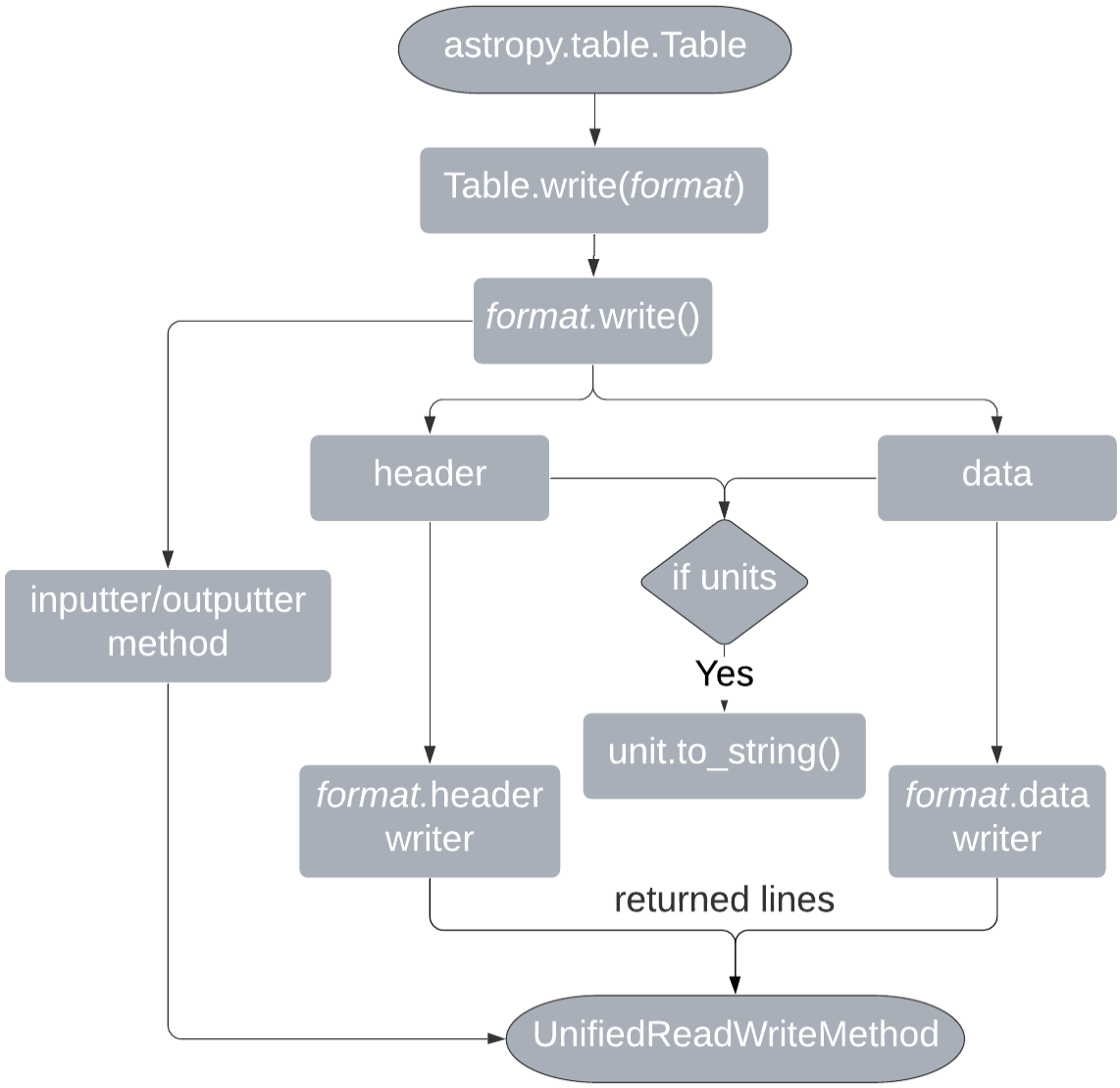
Developed in 1993, the CDS-ASCII file format solves these by separating the given input tabular data into two individual files.

I. A *description file* aka *ReadMe file* containing a description of the table.
II. *Plain ASCII file(s)* containing the actual table data.

The *ReadMe file* is essentially the key to the table and is an important characteristic of the CDS-ASCII format. It contains different sections describing different aspects of the table. What makes the *ReadMe file* different from FITS headers, which largely serve the same function, is that the *ReadMe file* is created for human readability and not solely for computer usage. As such, *ReadMe files* make it easier to compare tables. They also contain all the information necessary to convert the table to another format. [5]

The actual table data is stored in the form of *Plain ASCII files* which makes the overall size smaller and at the same time allowing their conversion to other formats without fuss using the information stored in the *ReadMe file.*

Writing in astropy



The File Processing in the Astropy package, i.e. reading and writing of tabular data files, is accomplished by the `astropy.io` module. This module has two sub-modules `astropy.io.fits` and `astropy.io.ascii` for handling FITS and ASCII format data tables respectively. ASCII formats include formats like HTML and LaTeX tables both of which have

both read and write methods available. A list of currently supported ASCII file formats in Astropy can be found at: https://docs.astropy.org/en/stable/io/ascii/#id3. Each supported ASCII format has a corresponding class defined for it in `astropy.io.ascii`, with methods for reading and writing files in that format. Presently, the CDS-ASCII data format class `astropy.io.ascii.Cds` only has the read method available as `astropy.io.ascii.Cds.read()`. This project will add a `astropy.io.ascii.Cds.write()` method to it, thus enabling writing files to the CDS-ASCII format.

The crucial entity for any Astropy input output operation is the `astropy.table.Table` object. Data being read from a file is stored into an Astropy `Table` and it is this `Table` object that is converted to another format and then written to a file. The accompanying flow chart on the previous page crudely illustrates the process of writing an `astropy.table.Table` object to any particular file format. Essentially, all the *`format`*`.write` methods rely upon the basic universal writer to finally write the table to a file. What each of the *`format`*`.write` methods do is convert the table into the designated format and return a list of lines that can be directly saved into a file. The `Table` object also has metadata associated with it in the form of a `meta` dictionary accessible through `Table.meta()`. This metadata is used to create the header for the file via the *`format`*`header` function of the concerned format. If the table contains any units then they are formatted using the `astropy.units` module. [2]

cds.pyreadme package

A typical CDS-ASCII format table contains the following sections in its *ReadMe file*:

- Title, authors, bibcode and other details.
- ***Keywords***, listing both data-related keywords and the printed publication related keywords.
- ***Description*** section, describing the context of the data, for example the instrumentation used or the observing conditions.
- ***File Summary***, describing the *name, length, number of lines* and *caption* of the file.
- ***Byte-by-Byte*** description of the file, specifying the *format*, *units*, *label* and *notes* for each column of the table.

CDS already has a package to create such a *ReadMe file* from any table including the generation of the ***Byte-by-Byte*** description of the file. It is available at cds-astro/cds.pyreadme and relies on `astropy.io.ascii` module for its purpose. This *cds.pyreadme* package can also build a Machine Readable Table (MRT) which is a concatenated form of the *Plain ASCII file* and the *ReadMe file* in a single table containing a header. The CDS-ASCII writer for Astropy can utilize the functionalities provided by the *cds.pyreadme* package to generate a *ReadMe file* for

any `astropy.table.Table` object in an straight-forward manner. The figure appended below, taken from *cds.pyreadme*'s repository, shows a template of a typical *ReadMe file.* [5]

CDS also has an accompanying program called *anafile* (http://vizier.u-strasbg.fr/doc/anafile.htx) which is meant to check whether or not a generated *ReadMe file* corresponds to the tabular data and to report any inconsistencies. Modifications and extensions based on *anafile* can be written in Python for checking the generated *ReadMe file* for any table. [1]

```
$catalogue                                                        ($author, $date)
================================================================================
$title
    $authors
    $bibcode
================================================================================
Keywords: $keywords

Objects:
    ----------------------------------------
       RA   (2000)   DE    Designation(s)
    ----------------------------------------

Abstract:
  $abstract

File Summary:
--------------------------------------------------------------------------------
 FileName    Lrecl   Records    Explanations
--------------------------------------------------------------------------------
$tablesIndex

--------------------------------------------------------------------------------
$bytebybyte

See also:
$seealso

Acknowledgements:

References:
================================================================================
     (prepared by author  / pyreadme )
```

Prior contributions to astropy

As part of my application for the GSoC 2021, I contributed to the Astropy library by opening the Pull Request (PR) #11508 on GitHub. This PR tries to solve issue #3972, that of adding a `_repr_latex_` method to `astropy.table.Table` object. I chose to work on this issue because it is highly connected to the topic of this proposal. Working on issue #3972 exposed me to the Astropy development environment and taught me the nuances of how different related methods in Astropy worked. I have been able to provide the following solution for the problem defined in #3972.

```
    def _repr_latex_ (self, latexdict={}):
        from astropy.io import ascii

        if latexdict in list(ascii.latex.latexdicts.keys()):
            latexdict = ascii.latex.latexdicts[latexdict]
        else:
            latexdict = self.meta.get('latexdict', {})

        lTable = ascii.latex.Latex(latexdict=latexdict)
        lines = lTable.write(table=self)
        self.meta['latexdict'] = lTable.latex

        return '\n'.join(lines)
```

As detailed in the previous section, the `latex.Latex.write` function returns the table in the form of a list of lines formatted in the asked data format. Using the default values of LaTeX formatting options, the `_repr_latex_` function given above returns a LaTeX formatted table as a string which can be directly used.

Deliverables

At present, there are three open issues in the Astropy repository on GitHub that relate to the topic of this proposal:

1. #9291 - CDS reader ignores nullability for columns with a limits specifier
2. #11239 - `astropy.io.ascii.Cds` format exception (`ValueError` for invalid `NULL`)
3. #11257 - Allow writing of Table to cds format

Out of these, the third issue is the main issue defining the primary aim of this project. The first two relate to compatibility checks on the `astropy.io.ascii.Cds` class. Such exception

handling and removal of bugs would also be important after the main issue is tackled and a CDS-ASCII writer in `astropy.io.ascii` is obtained.

Using the methods from *cds.pyreadme* package, it would be an easy extension to output a MRT file from an input `astropy.table.Table` once the CDS-ASCII writer for Astropy is ready. Since the *ReadMe file* of the CDS-ASCII data format is the header of a MRT file, the metadata of a `Table` can be used to convert one to the other. This is not a mentioned milestone for the project, but I would like to attempt it once the basic requirements of the project are in order.

The deliverable objectives at the end of the GSoC time period would thus be as follows:

- Proper understanding of the CDS-ASCII data format.
- The CDS-ASCII writer for `astropy.table.Table` written in Python.
- Tests for the newly added method.
- Proper documentation for the writer.
- Bug fixes, if any are reported.
- (*Maybe*) Feature to use *anafile* for checking the correspondence between *ReadMe file* and the data in the table.
- (*Maybe*) Feature for conversion of CDS-ASCII file to the MRT format.

Tentative Timeline

A tentative timeline for achieving the milestones and deliverable objectives of the project in the 10 weeks duration of GSoC 2021 is given below.

17 May – 7 June (Community Bonding Period)

- Get to know fellow developers contributing to Astropy.
- Talk to the Mentors.
- Learn more about open source software development and software releases.
- Explore what functionalities are provided by Astropy and how it is used by Scientists world-wide.

7 June – 21 June (Week 1&2)

- Understand the CDS-ASCII data format.
- Write an example CDS-ASCII writer for `astropy.table.Table` using Python.

21 June – 5 July (Week 3&4)

- Get familiar with the `astropy.io.ascii` framework.
- Test the example writer.

5 July – 19 July (Week 5&6, First Evaluations)

- Add the CDS-ASCII writer to `astropy.io.ascii` by opening a PR.
- Check if all the automated Astropy tests are passing.
- Add documentation for the newly added writing method.

This PR will close the main issue, #11257, relating the project and the milestone for First Evaluations will be achieved.

19 July – 2 Aug (Week 7&8)

- Finishing running tests on the writer.
- Work on the complementary issue #9291, if it isn't closed yet.
- Complete the documentation.

2 Aug – 16 Aug (Week 9&10)

- (*Maybe*) Add feature to use *anafile* for checking the correspondence between *ReadMe file* and the data in the table.
- Work on the complementary issue #11239, if it isn't closed yet.
- Fix any bugs reported.
- Rerun tests and update the documentation.

Hopefully, by now almost all of the project objectives would be achieved with proper documentation for the new features. The code would thus be ready for submission and final evaluations.

16 Aug – 30 Aug (Final Evaluations)

- (*Maybe*) Add feature for converting the CDS-ASCII written file to MRT format.
- Submit the code and documentation for the Final Evaluations.

September and beyond

By this the project would have come to an end. If possible, I would like to present the work completed in this project in some conference, like the next PyCon. I also intend to continue contributing to Astropy by working on other open issues and reviewing PRs.

Conclusion

The project proposal detailed in these pages focuses on adding a CDS-ASCII format writer to the Astropy library. Reasons for why such a feature would be useful and how the writer can be implemented in the present Astropy architecture have been described. Astronomical data catalogs derived from published materials are largely stored in the CDS-ASCII format. Addition of the features and methods discussed in this proposal will provide a handy tool for researchers and scientists submitting their data to astronomical journals. Given my background in Astrophysics and my computational skills, I believe I am an apt candidate for working on this project as part of the GSoC 2021 program.

References

[1] *anafile*. (2007, January 7). Retrieved April 12, 2021, from http://vizier.u-strasbg.fr/doc/anafile.htx

[2] The Astropy collaboration. (2018, August 24). The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. *The Astronomical Journal*, *156*(123). https://doi.org/10.3847/1538-3881/aabc4f

[3] *Centre de données astronomiques de Strasbourg*. (n.d.). Retrieved April 12, 2021, from https://www.wikiwand.com/en/Centre_de_donn%C3%A9es_astronomiques_de_Strasbourg

[4] *Preparing and Submitting Tabular Data*. (n.d.). Retrieved April 12, 2021, from http://vizier.u-strasbg.fr/vizier/doc/submit.htx

[5] *VizieR catalogue - The Question of Standards*. (n.d.). Retrieved April 12, 2021, from http://vizier.u-strasbg.fr/doc/catstd-1.htx

---