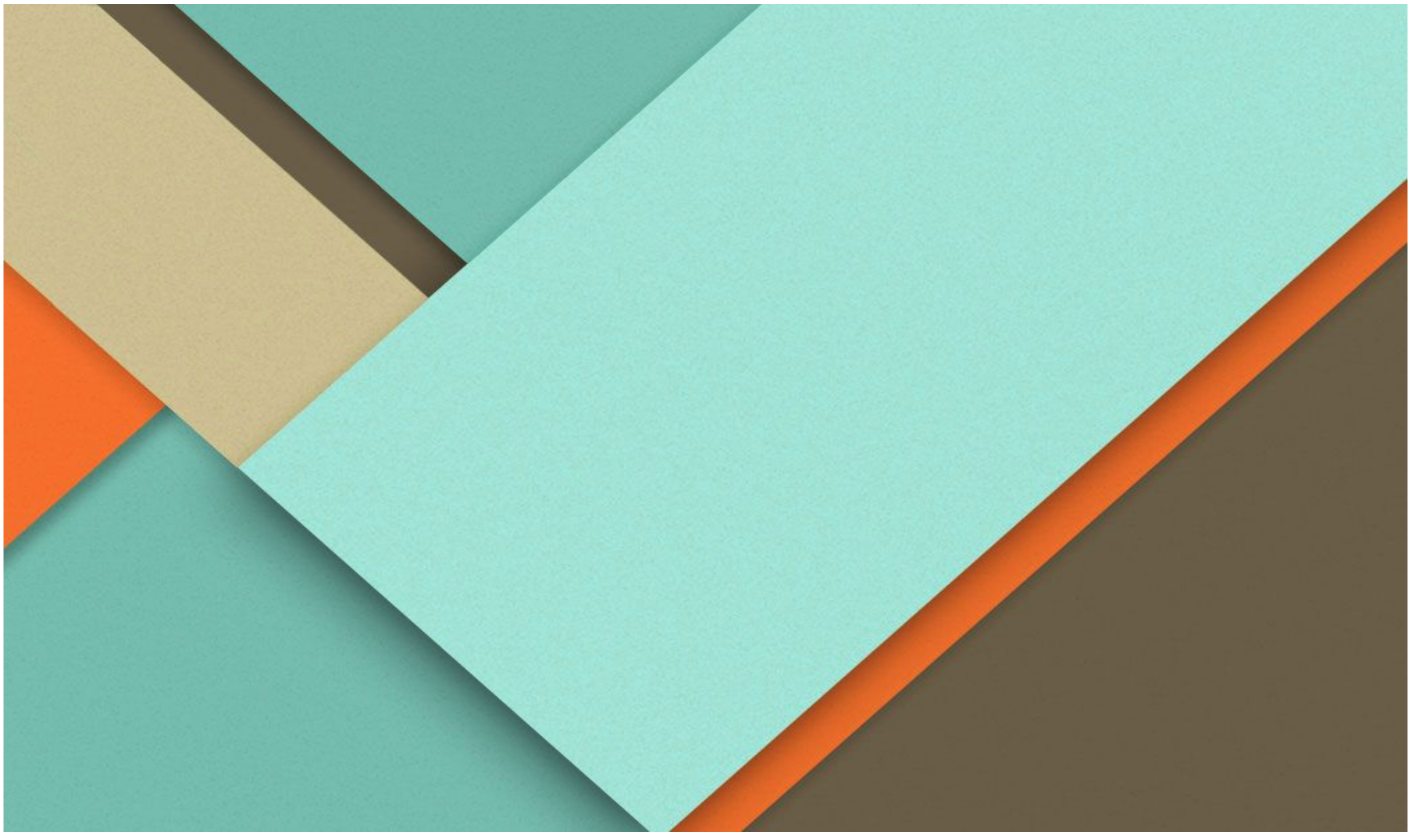

# Supporting APE14 in NDCube

Google Summer of Code 2019

Organisation: Open Astronomy

Sub Organisation: Sunpy (NDCube)

Mentors: Daniel Ryan, Stuart Mumford

# About Me

## Basic Information

Name - Yash Sharma
University - IIT Kharagpur
Major  - Integrated BS-MS in Mathematics and Computing
Github - `yashrsharma44`
Riot-ID - `hitman23`
Blog - Medium
Timezone - IST (UTC +5:30)
Contact Email - `yashrsharma44@gmail.com`
Contact Number - +91-859-716-5704

## Personal Background

I am a second year undergraduate of the Department of Mathematics, enrolled in its Integrated MSc. course at Indian Institute of Technology, Kharagpur. I use Ubuntu 16.04 LTS as my regular operating system with VSCode as my primary text editor. I am proficient in C/C++, Java and Python programming languages.

My predominant programming is done with Python because it easily helps me converting my ideas into code. Python has a user-friendly approach to programming while providing a solid interface to OOP concepts very well.

## Contributions/Pull Requests in Sunpy/NDCube

- Mocking `sunpy.net.downloader` -
  1. Mocking `test_noaa.py - [Merged] [PR 2900]`
  2. Mocking `test_norh.py - [Open] [PR 2923]`
  3. Mocking `test_lyra_ud.py - [Open] [PR 2958]`
  4. Mocking `test_eve.py - [Open] [PR 2961]`
  5. Mocking `test_goes_ud.py - [Open] [PR 2962]`
- Added support for writing FITS file from NDCube - `[Open] [PR 154]`

## Approach to the project

NDCube is a SunPy-affiliated package for generalized handling, manipulating and visualizing N-dimensional astronomical data. Currently, the translations between the pixel indices and the real world coordinates are described by the FITS-standard World Coordinate System (FITS-WCS). However, other implementations of WCS also exist, such as generalized WCS (gWCS) developed by astropy. In order to support gWCS, and other WCSs in the near future, an API was proposed, in which all such WCSs were required to propose a given set of `API endpoints`, thereby facilitating the use of different types of WCSs that exist, and abstracting the inner methods, without adding overhead to work with these methods.

This document proposes to add the support for APE14 in NDCube, which is a SunPy-affiliated project. In order to support both FITS-WCS and gWCS and hence support more future WCSs libraries, this project aims to convert the ndcube package to use a common WCS API. The new API has already been outlined by astropy's `APE14`. Implementing support for APE14 will enable ndcube to use FITS-WCS and gWCS independently and hence increase the power and scope of the ndcube package. With this new feature, NDCube will become better placed to serve a wider array of n-dimensional data analysis needs from multiple astronomical communities.

### Breakdown of the project

The document breaks down the deliverables into the following parts -

#### `Use APE14 methods in NDCube`

This part of the project will make sure that the NDCube now uses the APE14 methods instead of the old methods of the current FITS-WCS object from astropy. This would make the NDCube ready for using any WCS independent methods. The `wcs.py` , `ndcube.py`, `ndcube_sequence.py`, `ndslicing.py`, `plotting.py` and `sequence_plotting.py` along with other methods will undergo change, implementing APE14 methods. For this, we use the `astropy.wcs.WCS` which contains all the methods defined in the APE14 for supporting FITS-WCS.

#### `Abstraction of slicing in NDCube`

After APE14 methods, the abstraction of slicing needs to be done. A `classmethod` can be proposed, which would take in a wcs object, and then perform the requisite slicing. After the method is made, the FITS-WCS object would be supported through this abstraction. This would be carried out with requisite tests and documentation, which would explain the working of the code. There is a subsequent `WIP PR` going on in Astropy, which can be used for generalizing the slicing. Since the Pull Request is open, I would try to contribute to it, while utilizing it in `NDCube` to support slicing.

`Reimplement the ndcube visualization mixins to use APE 14.`

Since the visualizing part of the project consists of various attributes of the WCS object, so this needs to be updated with APE14 methods. Corresponding tests and documentation needs to be carried out, also we would ensure that the visualizing works for any WCS object. Current implementation would ensure that visualization of FITS-WCS is working as it was working earlier.

## Timeline of the Project

### Community Bonding Period (May 6 - May 27)

During the community bonding period, I would start discussing with my mentors, about any change in the proposal. I would be participating in a research project with my university's professor, so I would be fairly busy. I would be free by May 20, so that should not be a hurdle, given that the coding period begins from May 27. I would be closely looking at the codebase of NDCube and would start to prototype the proposed API from May 20. I have fair experience with pytest and mocking, so that should be up and running for me. Other than that, I would like to solve an issue or two of NDCube, if time permits.

At the end of the Community Bonding Period, I would aim to -

- Get familiar with the codebase of NDCube
- Start with prototyping the proposed API
- Solving an issue / complete working on my `[PR]`

### Week 1 (May 27 - June 2)

- Start discussing the shortcomings of the prototype of the API
- Make the permanent changes to code-base by sending a WIP PR
- Implement APE14 methods in `wcs.py`
- Documentation and tests for the same
- Tests would be written which would be `WCS` agnostic and would work for any `WCS` object supporting `APE14` methods.

### Week 2 - Week 3 (June 3 - June 16)

- Implement APE14 methods in `ndslicing.py`, `plotting.py`, `ndcube.py` and `ndcube_sequence.py`.
- Documentation and tests for the same

### Week 4 - Week 5 (June 17 - June 23)

- Implement APE14 methods in `sequence_plotting.py` and cover up any leftover methods.
- Documentation and tests for the same

### Phase 1 Evaluation

### Week 6 - Week 7 (June 25 - July 5)

- Start with the abstraction of slicing in WCS.
- Start coding up the `classmethod` for slicing.
- Discuss with mentors with the progress and possible hurdle
- Get the previous PR for Phase 1 merged, if not done.
- Documentation and tests for the same

### Week 8 (July 6 - July 12)

- Complete up with the abstraction of slicing
- Try up various edge cases for testing out the slicing the WCS object
- Documentation and tests for the same

### Week 9 (July 13 - July 21)

- Buffer week for leftover tasks.
- Discuss with the mentors about the visualization mixins, and start with a prototype.
- Get the PR merged into master

### Phase 2 Evaluation

### Week 10 - Week 11 (July 23 - August 6)

- Start with the previous work of using APE14 in the `plotting.py` and `sequence_plotting.py`.
- Testing out the different edge cases for the plotting.
- Documentation and tests for the same

### Week 12(August 7 - August 14)

- Continue and finish up with the implementation of the `visualization mixins`.
- Finish up the plotting and send up a polished PR for review
- Leftover tests and documentation

### Week 13(August 15 - August 25)

- Write up documentation for leftover tasks
- Write documentation about the usage of the project.
- Discuss with mentors about shortcomings.
- Small nit-pickings about the project.
- Complete leftover tasks, if any.

### Final Evaluation

# GSOC

## Q) Have you participated previously in GSoC? When? With which project?

Yes, in GSoC 2018. Under Scrapinghub Organisation, under Python Software Foundation

## Q) Are you also applying to other projects?

No, I am not applying to any other projects in any other organisation. I am applying to SunPy organisation and this is the only project, I am applying for.

## Q) Commitment

I will be participating in a research project for 20 days from May 1 to May 20. Since this period lies in the Community Bonding Period, I think it does not affect my coding period, which starts from May 27.

My vacations start from May 1 and last until July 15. So I would be free during the vacations and can contribute to 40 hours of work/week.

My classes start from July 16, however, I can manage time for 35+ hours/week as, during the start of the semester, there is less academic load.

## Q) Eligibility

Yes, I am eligible to receive payments from Google. For any queries, clarifications or further explanations, feel free to contact `yashrsharma44@gmail.com`