Siddharth Lal

siddharthlal25@gmail.com
github.com/siddharthlal25
+91 799 113 8037

Astronomical Data Reduction and Processing

Mentors- Miles Lucas, Mosè Giordano, Yash Sharma

Google Summer of Code 2020 | Open Astronomy | Julia Astro

OVERVIEW

GOALS

- Algorithms
- Data Structure
- Iterator Reduction
- Stretch Goals

TIMELINE

- Time Commitments
- Weekly Details

ABOUT ME

- Basic Information
- Platform Details
- Background
- Contributions

OVERVIEW

Over the years Julia has become a successful language for science and numerical computing. However, there are no current packages for performing CCD image reduction, which is essential for any observational astronomer. As a part of this project, my goal is to solve this problem by developing a new package for CCD data reduction, which will be easy to use, clairvoyant in its effects, and provide comparable utility to other reduction packages.

GOALS

The goal is to create a new library for astronomical data reduction in Julia, the primary aim being to implement the basic functions for data reduction followed by creating an elegant data structure to store the CCD data with corresponding supporting data, such as uncertainty, units and metadata corresponding to image data. The last stage would be to develop a pipeline for the complete reduction process.

ALGORITHMS

The algorithms I will implement are:

1. Trimming
2. Combination
3. Overscan Subtraction
4. Bias Subtraction
5. Dark Subtraction
6. Flat Fielding

After the implementation of these algorithms, the focus will be on developing a data structure and a pipeline for complete data reduction along with thorough documentation and extensive tests. The development of the complete package will be done keeping in mind its integration with *Unitful.jl*. Below is a small description of algorithms to be implemented.

TRIMMING

This feature will allow morphological changes in the image, such as removing side rows or columns. This would inherently be a mutating function. The function call would appear as:

```
trim(img, 2049:lastindex(img,1), 2049:lastindex(img,2))
trim(img, "[1:2048, 1:2048]")
```

This trims the **`img`** named image to the mentioned dimensions. This is functionally identical to cropping.

COMBINATION

One can combine many images to lower their standard error, as well as removing transient outliers, such as cosmic rays. There are various methods that can be applied for combining.

A commonly used combining method is median stacking, where a median image is generated from a set of images and used for further processing.

OVERSCAN SUBTRACTION

Professional CCDs often contain regions of unexposed CCD that are still read out. These regions are called “overscan” regions. This region can be used to estimate the bulk offset, mentioned above. Often this offset is actually estimated via a low-order polynomial fit to whichever pixels are in the overscan region. This constant offset value is then subtracted from any image. Note that this process occurs completely within a single frame (i.e. there is no extra calibration file needed to perform overscan subtraction).

BIAS SUBTRACTION

This method deals with removing bias that is caused due to the constant voltage applied to the CCD detector. The bias frame is captured by taking a covered exposure at the fastest possible shutter speed. Bias frame is meant to capture the instantaneous pixel-by-pixel noise of the detector, which consists of three main components. The first is a bulk offset. This value is constant, therefore it has no noise. The second is pixel to pixel variations in the bulk offset. This is called our bias signal. Lastly, there is read noise from propagating the signal from the CCD through the electronics.

For reduction, many bias frames are taken and a master bias frame is created by subtracting the overscan value and trimming the frame to remove the overscan region for each frame, then median combining all of the frames.

DARK SUBTRACTION

Dark signal is a signal accumulated from thermal effects on the CCD over time. They need to be taken at the same temperature and for the same exposure time as the science frames. Most professional CCDs skip this step because the high-precision cryogenic cooling of the CCDs removes nearly all thermal noise from the detector.

A master dark is created by median combining many darks that have been overscan corrected, trimmed, and bias-corrected.

FLAT FIELDING

This method deals with removing the effect of dust between source and camera, as well as variations in pixel sensitivity. Dust and sensitivity variation convolute the incoming signal and must be normalized to retrieve accurate photometry of sources. This issue can be corrected by recording/clicking several images (called flat fields) of spatially uniform

illuminated targets and median combining them. Two common sources are a white illuminated panel, which is invariant to spectral aberrations and short exposure of a section of the sky without any visible stars, which will be variant to spectral aberrations. Flat fields are required to be taken for each photometric filter which is being used in the process of capturing astronomical images.

To correct an image, first the flats need to be calibrated. Initially, they have their overscan subtracted and trimmed followed by subtracting the master bias frame. A master flat is formed by median combining the calibrated flats. Then, this flat is normalized to the target image's mean value and the target is divided by the master flat.

DATA STRUCTURES

The next aim is to design an elegant data structure to store the CCD data which supports features such as reading and writing data to FITS file format. This format is a binary hierarchical data format endemic to astronomy. They consist of header data units (HDU) that can contain array-like data or tabular data as well as a dictionary-like header with useful meta information. Using FITSIO.jl I can easily interface with FITS files, but I'd like my own type for dispatching across the package in an easy manner.

After the implementation of the above features, the skeleton of the package would allow us to perform operations as demonstrated below.

The data will be loaded as follows:

```
bias = CCDDataType("bias.fits")
flat = CCDDataType("flat.fits")
science = CCDDataType("science.fits")
```

For reduction, every bias frame is subjected to overscan subtraction followed by trimming and finally median combination is performed to obtain the master bias frame.

A single bias frame would be subjected to overscan subtraction and trimming as follows:

```
bias_ = trim(subtract_overscan(bias, "[2049:2068,1:2068]"),
"[1:2048,1:2068]")
```

Following this bias frames will be median combined resulting in the creation of a master bias frame (**mbias**).

The corrected flat can be generated by overscan subtraction on flat frames followed by bias subtraction with the master bias frame on it as follows:

```
flat_=subtract_bias(trim(subtract_overscan(flat,"[2049:2068,1:2068]"),
"[1:2048,1:2068]"),mbias)
```

Following this flat frames will be median combined resulting in the creation of a master flat frame (**mflat**).

This will be followed by overscan subtraction of **science** resulting in **science_** as follows:

```
science_ = trim(overscan_correct(science, "[2049:2068,1:2068]"),
"[1:2048,1:2068]")
```

Finally, the callibrated frames can be obtained by correcting the modified science (**science_**) with master bias (**mbias**) and master flat (**mflat**), in the following manner:

```
calibrated = flat_correct(subtract_bias(science_, mbias), mflat)
```

Following this, the calibrated frames would be median combined resulting in the creation of a final calibrated image after reduction.

ITERATOR REDUCTION

Astronomers deal with multiple files because of the different calibration frames and the fact that multiple images of each type need to be taken to reduce the standard error. This is why there is a need for a convenient way of iterating and performing operations over a set of data. This is done with the help of iterator reduction. It deals with making the functions more user friendly. It corresponds to two parts, accessing data from directories and sorting them on the basis of header files, and performing some kind of reduction as required on the data.

For example, the code snippet given below gives us an idea of how things will look after the implementation of this feature. The example is focused on creating a master bias from multiple bias frames.

```
collection = FileCollectionType("data")
bias_frames = filter(row -> row.imagetype == "BIAS", collection)
reduced_frames = []
for frame in ccds(bias_frames)
            bias_ = subtract_overscan!(frame, :BIASSEC)
            bias_ = trim(frame, :DATASEC)
            push!(reduced_frames, bias_)
end
master_bias = combine(reduced_frames)
write("master_bias.fits", master_bias)
```

First, the data is loaded followed by filtering on the basis of header data and then routine on each bias frame. Next, reduced bias frames are combined and dumped as a FITS file.

STRETCH GOALS

- Development of a separate package for image registration.
- Implementation of cosmic ray detection algorithm *LACosmic*.
- Implementation of non-linearity correction for CCD data.
- Implementation of various image statistics.

TIMELINE

TIME COMMITMENTS

Since my college semester ends on the 30th of April and the next semester starts around the end of July, I will have plenty of time to work on the project. Since I do not have any other commitments, I can comfortably spend 40 hours every week working on the project, and can easily bump it to 45 or even 50 if necessary for any particularly difficult or unexpected problems. Although the last couple of weeks of the coding period will clash with my college classes, since there will be no exams or any major commitments, I am confident I will be able to handle it without affecting my productivity. I will also be writing a fortnightly blog about the progress of my project. Every week I will also be opening a WIP PR for weekly goals that would help the mentors to track the progress.

WEEKLY DETAILS

Week 1 & 2

Implementation of trimming and combination followed by documentation and tests.

Week 3 & 4

Implementing overscan subtraction and bias subtraction, followed by documentation and tests.

Week 5

Buffer week to complete leftover if left otherwise proceed with the ahead timeline.

Week 6 & 7

Implementation of dark subtraction, flat fielding and creation of an elegant data structure to store the CCD data, followed by documentation and tests.

Week 8 & 9

Implementation of iterator reduction for CCD data, followed by documentation and tests.

Week 10 & 11

Integration of a pipeline for a complete subroutine of astronomical data reduction, followed by documentation and tests. The focus in these weeks would be given to writing notebooks to depict various examples of complete reduction for the end user.

Week 12

Buffer week to complete leftover work if left otherwise work on stretch goals.

ABOUT ME

Basic Information

Name	Siddharth Lal
University	Indian Institute of Technology, Kharagpur
Major	Mathematics and Computing
Email address	siddharthlal25@gmail.com
Github	siddharthlal25
LinkedIn	siddharthlal25
Contact Number	+91 799 113 8037
Timezone	IST (UTC +05:30)

Platform Details

OS	Ubuntu 18.04
Editor	Atom

Background

I am a third-year undergraduate student at the Indian Institute of Technology, Kharagpur majoring in Mathematics and Computing. I was introduced to the world of programming through Visual Basic as a part of my middle school curriculum. What started as a school course soon became my favorite hobby. Eventually, I was introduced to algorithms and the clean and elegant methods with which they solved all sorts of problems captivated me. This also led me to pick up competitive programming as a hobby which I enjoy tremendously.

I was introduced to Julia around a year ago and was immediately amazed at how simple yet powerful the language is. The elegant syntax made it very easy to learn and the comprehensive documentation ensured that nothing was left uncertain. I have enjoyed learning and working with the language immensely, and would love to use GSoC as an opportunity to contribute using Julia.

I have a fairly strong background in mathematics and physics, also I have taken courses on linear algebra, matrix algebra, statistics, and some basic level physics courses. I am also currently ranked fourth in my department with a cumulative GPA of 9.42/10. Coupled with my knowledge of mathematics, physics and eagerness to learn the intricacies of astronomy, I feel I am sufficiently well-versed in all the relevant fields to successfully bring this project to fruition.

Contributions to Julia Astro

Since my project involves the creation of an entirely new package in Julia from scratch, I decided to contribute to a package related to astronomy in Juliastro. The contributions are as follows:

PRs:

1. Added Elliptical Aperture: https://github.com/JuliaAstro/Photometry.jl/pull/4
2. Added Elliptical Annulus: https://github.com/JuliaAstro/Photometry.jl/pull/8
3. Removed redundant calls: https://github.com/JuliaAstro/Photometry.jl/pull/7
4. Added sigma clipper: https://github.com/JuliaAstro/Photometry.jl/pull/14
5. Added Median statistic: https://github.com/JuliaAstro/Photometry.jl/pull/15
6. Added Mode statistic: https://github.com/JuliaAstro/Photometry.jl/pull/16
7. Added MMMBackground: https://github.com/JuliaAstro/Photometry.jl/pull/18
8. Added SourceExtractor: https://github.com/JuliaAstro/Photometry.jl/pull/19
9. Added BiweightLocation: https://github.com/JuliaAstro/Photometry.jl/pull/21
10. Added IDW Interpolator: https://github.com/JuliaAstro/Photometry.jl/pull/26

There were a few docs fixing PR's which are not mentioned over here.

Issues:

1. Redundant calling of bbox: https://github.com/JuliaAstro/Photometry.jl/issues/6
2. Adding units: https://github.com/JuliaAstro/UnitfulAstro.jl/issues/16