# SMARTScience Tools: Interacting With Blazar Data Dynamically

*Imran Hasan, Charles Bailyn, Victoria Misenti, Jedidah Isler, Meg Urry, Emily McPherson, Michelle Buxton, Paolo Coppi*

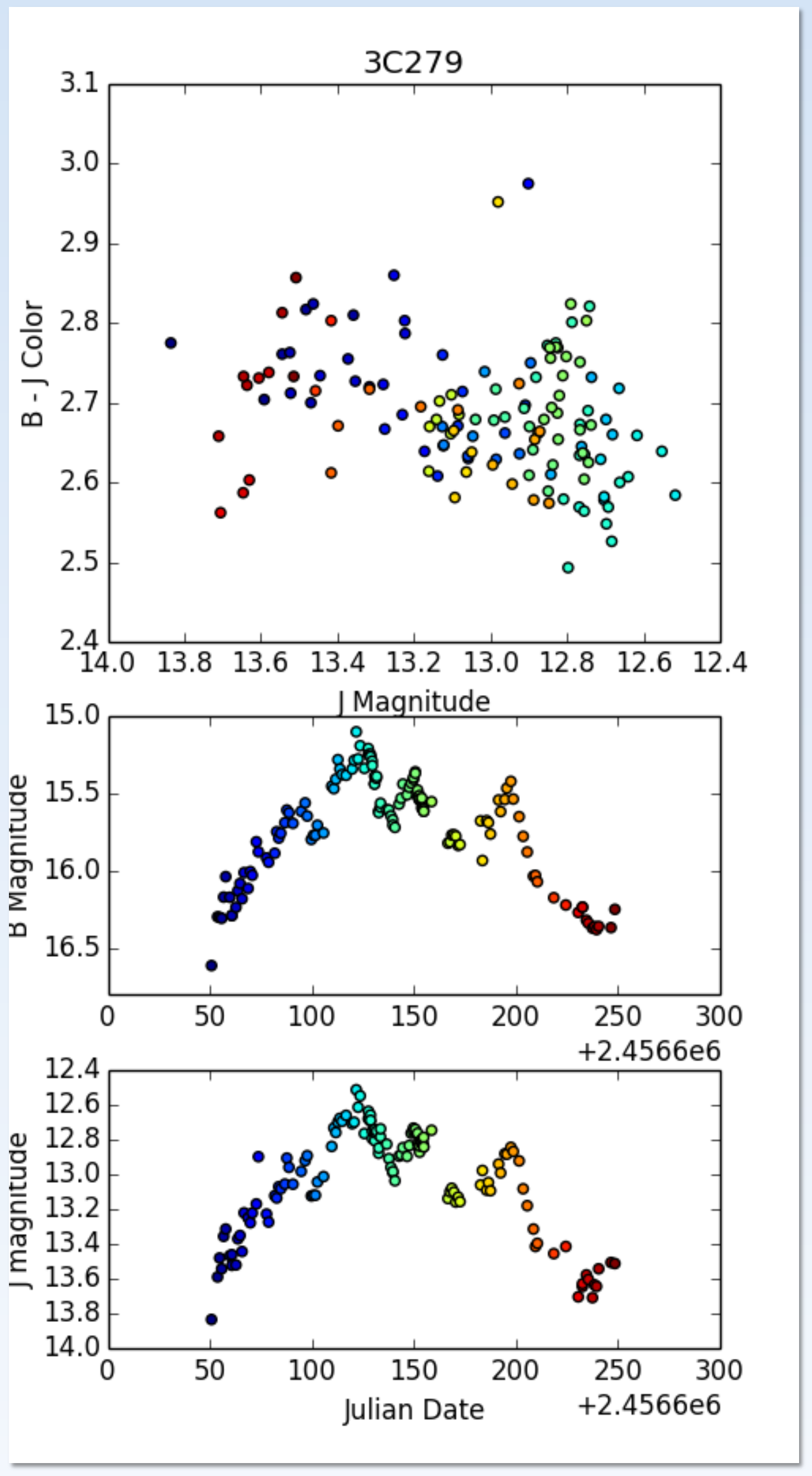
*Yale University*

## Motivation

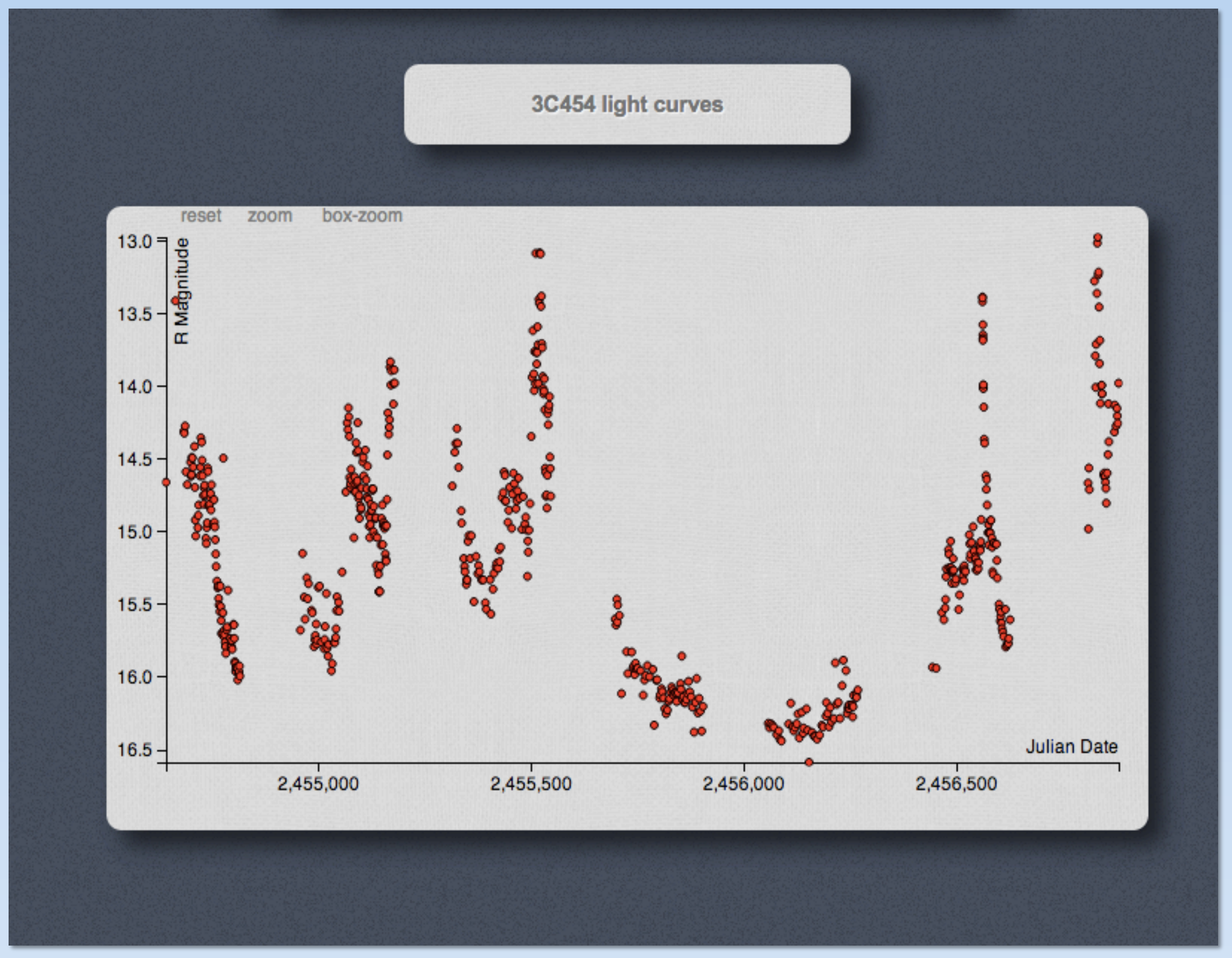
The Yale-SMARTS blazar group has used SMARTS telescopes and ANDICAM, located at CTIO, for over 6 years to gather optical-IR photometry of more than 70 Fermi-detected blazars. To enable public visualization and study of this data set, we developed SMARTScience Tools, which allows users to intuitively and easily dynamically interact with the data. SMARTScience Tools significantly improves the public's ability to use the Yale-SMARTS data set, and can make multiwavelength studies of blazars more accessible, efficient, and community driven.

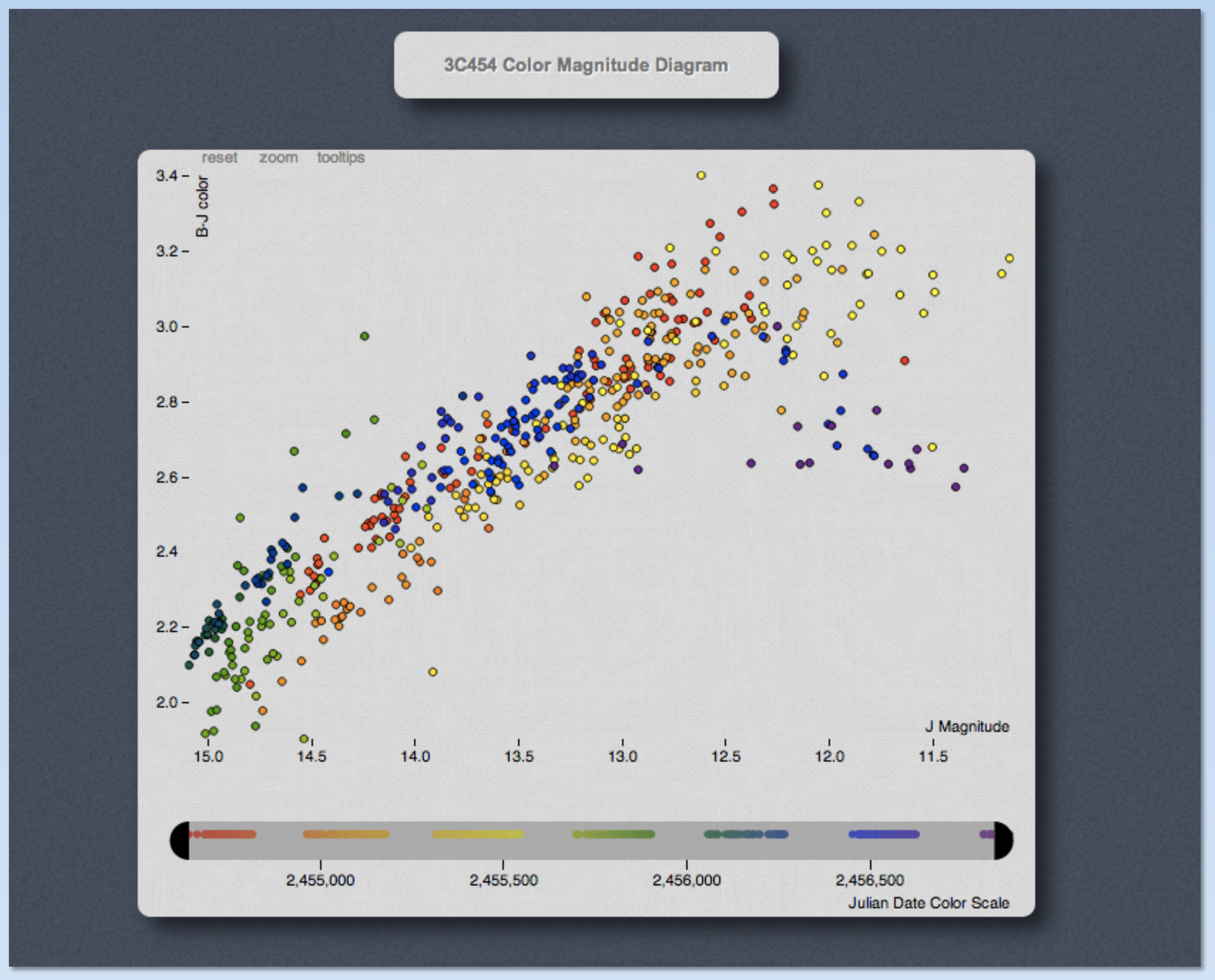
## Science Objectives

Yale-SMARTS data is showing a relation in the color-magnitude evolution over the course of optical-IR flares. Be sure to check out Jedidah Isler's presentation for more. SMARTScience Tools will allow users to discover these kinds of relations in targets quickly and easily.



## On The Web

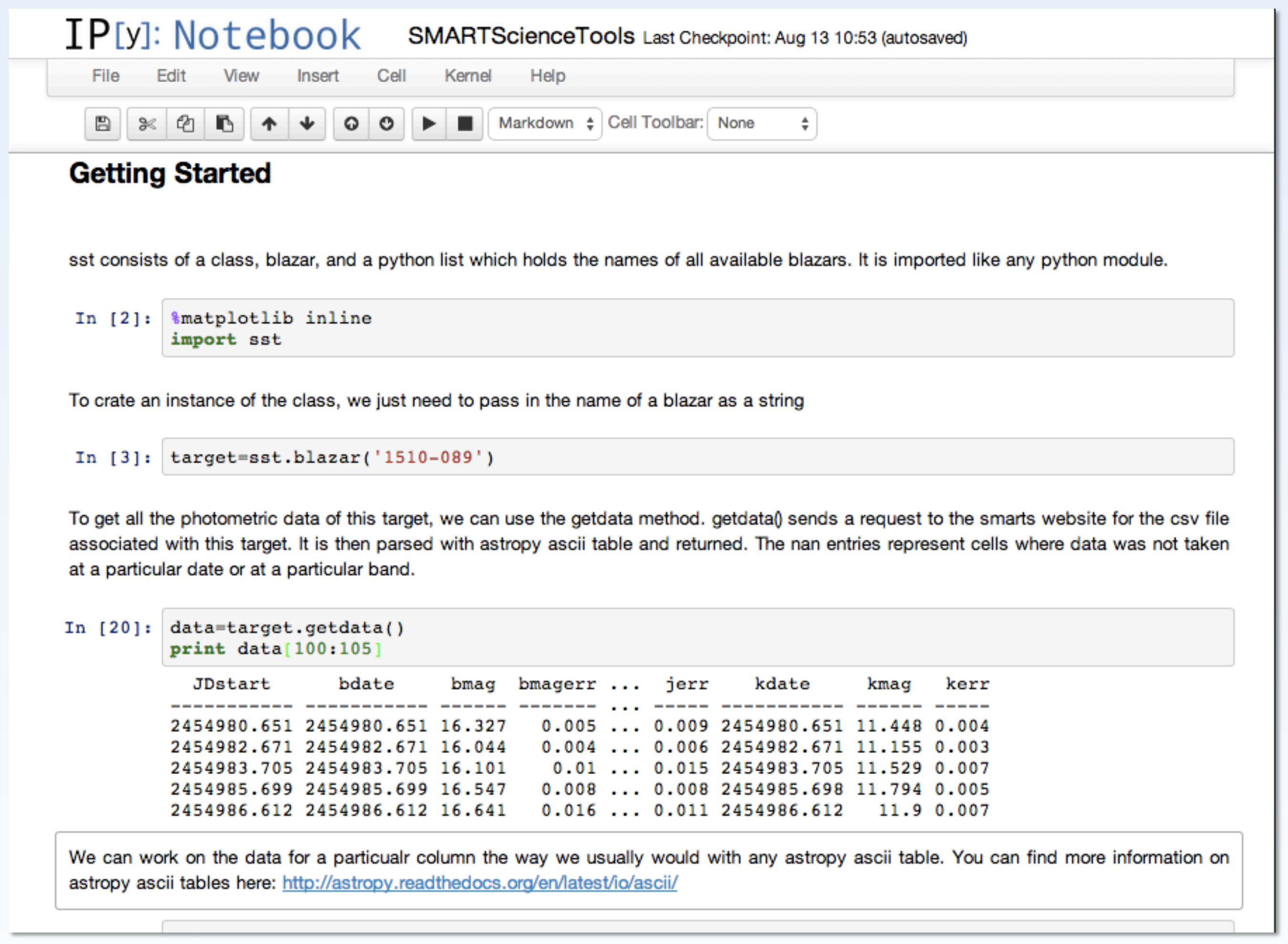




Users are able to create customizable light curves and color magnitude diagrams. Specific bands can be selected to construct plots, which include features such as dynamic zooming, panning, and direct mouse interaction with individual points. Additionally, human- or machine-readable tables of the data can be printed directly for further independent study by the user. For further convenience, nearly all of these features can be used on mobile devices. Try it at bit.ly/SMARTScienceTools.

## In Your Terminal

A small API written in Python allows users to get and work with the most up to date data. Using the API, users can query the Yale-SMARTS website directly for data, and parse it into an astropy ASCII table. The API also leverages matplotlib to create quick light curves and color magnitude diagrams. A tutorial in an ipython notebook is available here: bit.ly/SMARTScienceTutorial.



SMARTScienceTools Last Checkpoint: Aug 13 10:53 (autosaved)

**Getting Started**

sst consists of a class, blazar, and a python list which holds the names of all available blazars. It is imported like any python module.

```
In [2]: %matplotlib inline
        import sst
```

To crate an instance of the class, we just need to pass in the name of a blazar as a string

```
In [3]: target=sst.blazar('1510-089')
```

To get all the photometric data of this target, we can use the getdata method. getdata() sends a request to the smarts website for the csv file associated with this target. It is then parsed with astropy ascii table and returned. The nan entries represent cells where data was not taken at a particular date or at a particular band.

```
In [20]: data=target.getdata()
         print data[100:105]

  JDstart      bdate      bmag  bmagerr ... jerr    kdate      kmag   kerr
----------- ----------- ------ ------- ... ----- ----------- ------ -----
2454980.651 2454980.651 16.327   0.005 ... 0.009 2454980.651 11.448 0.004
2454982.671 2454982.671 16.044   0.004 ... 0.006 2454982.671 11.155 0.003
2454983.705 2454983.705 16.101    0.01 ... 0.015 2454983.705 11.529 0.007
2454985.699 2454985.699 16.547   0.008 ... 0.008 2454985.698 11.794 0.005
2454986.612 2454986.612 16.641   0.016 ... 0.011 2454986.612   11.9 0.007
```

We can work on the data for a particualr column the way we usually would with any astropy ascii table. You can find more information on astropy ascii tables here: http://astropy.readthedocs.org/en/latest/io/ascii/

## References

This work is made possible by software from d3.js, Github, matplotlib and astropy. The Yale-SMARTS Blazar program is funded by a grant from the Fermi Special Guest Investigator Program.