SunPy 0.8: Python for Solar Physics

Monica G. Bobra[1], Andrew R. Inglis[2], Stuart Mumford[3], Steven Christe[4], Nabil Freij[5] , Russell Hewett[6], Jack Ireland[7], Juan Carlos Martínez Oliveros[8], Kevin Reardon[9], Sabrina Savage[10], Albert Shih[4], David Pérez-Suárez[11], and the the SunPy Community

1. Stanford University, Stanford, CA, USA, 2. Catholic University of America, Washington, DC, USA, 3. University of Sheffield, Sheffield, UK, 4. NASA Goddard Space Flight Center, Greenbelt, MD, USA, 5. Universitat de les Illes Balears, Palma, Spain, 6. Total SA, Houston, TX, USA, 7. ADNET Systems Inc., Greenbelt, MD, USA, 8. University of California, Berkeley, CA, USA, 9. National Solar Observatory, Tucson, AZ, USA, 10. NASA Marshall Space Flight Center, Huntsville, AL, USA, 11. Mullard Space Science Laboratory, Dorking, UK.

SunPy is a community-developed open-source software library for solar physics. It is written in Python, a free, cross-platform, general-purpose, high-level programming language which is being increasingly adopted throughout the scientific community. SunPy aims to provide the software for obtaining and analyzing solar and heliospheric data. This poster introduces a new major release, SunPy version 0.8. The first major new feature introduced is Fido, the new primary interface to download data. It provides a consistent and powerful search interface to all major data providers including the VSO and the JSOC, as well as individual data sources such as GOES XRS time series. It is also easy to add new data sources as they become available, i.e. DKIST. The second major new feature is the SunPy coordinate framework. This provides a powerful way of representing coordinates, allowing simple and intuitive conversion between coordinate systems and viewpoints of different instruments (i.e., Solar Orbiter and the Parker Solar Probe), including transformation to astrophysical frames like ICRS. Other new features including new timeseries capabilities with better support for concatenation and metadata, updated documentation and example gallery. SunPy is distributed through pip and conda and all of its code is publicly available (sunpy.org).

Scientific Python

Programming in Python allows users to utilize a large and growing ecosystem of scientific packages already in place, including:

NumPy – for fast array operations
SciPy – for common scientific operations, e.g. curve fitting
Pandas – for time series manipulation
AstroPy, **SpacePy**, **PlasmaPy** – for astrophysical tools
Scikit-image, **opencv**, **pillow** – for image processing and computer vision
Scikit-learn, **caffe** – for machine learning

Get Involved

There are a number of ways to make your voice heard! We actively solicit code development and feedback from the community. We pride ourselves on being a welcoming community and we would love to have you become a part of this!

To get started, visit our website at **sunpy.org** or check out our documentation at **docs.sunpy.org.** If you're interested in chatting with us, you can mail the general SunPy mailing list at **sunpy@googlegroups.com** or find us in our chat room at **matrix.org**.

Finally, come develop code or otherwise contribute to SunPy! The people who help develop or contribute to SunPy are varied in ability and experience. The Google Summer of Code (GSOC) and the European Summer of Code in Space (SOCIS) programs provided funding for over a dozen summer students to work on SunPy development since 2011.

SunPy 0.8 – what's new?

We just released SunPy 0.8! The development of SunPy is a continuous, community-led process. For the latest release, there were 30+ contributors, with over 1300 commits to the codebase, addressing more than 160 issues. New headline features in 0.8 include:

Fido – the unified downloader

The new package Fido is the primary interface to search for and download observational data irrespective of the underlying client or web service through which the data is obtained. Here is an example query:

```
result = Fido.search(a.Time('2011-06-07 06:20','2011-06-07 06:30'),
                     a.Instrument('AIA') | a.Instrument('EUVI') |
                     a.Instrument('GOES'))
```

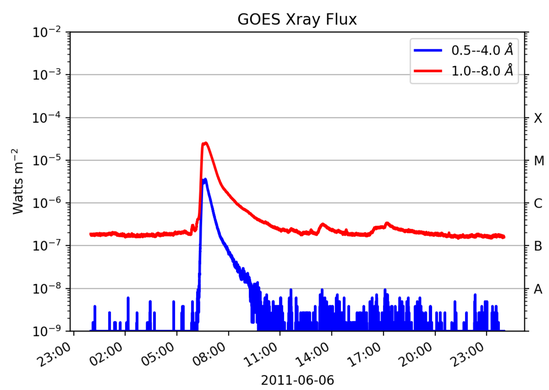
```
print(result)
```

```
402 Results from the VSOClient:
   Start Time [1]       End Time [1]     Source ...   Type     Wavelength [2]
                                            ...                    Angstrom
        str19                str19          str3  ...   str8        float64
------------------- ------------------- ------ ... -------- ----------------
2011-06-07 06:20:00 2011-06-07 06:20:01    SDO ... FULLDISK    171.0 .. 171.0
2011-06-07 06:20:00 2011-06-07 06:20:01    SDO ... FULLDISK    211.0 .. 211.0
2011-06-07 06:20:02 2011-06-07 06:20:03    SDO ... FULLDISK     94.0 .. 94.0
22 Results from the VSOClient:
   Start Time [1]       End Time [1]     Source  ...   Type    Wavelength [2]
                                            ...                    Angstrom
        str19                str19          str8  ...   str8        float64
------------------- ------------------- -------- ... -------- --------------
2011-06-07 05:03:55 2011-06-07 09:50:15 STEREO_B ... FULLDISK 171.0 .. 175.0
2011-06-07 06:20:10 2011-06-07 06:20:12 STEREO_B ... FULLDISK 171.0 .. 175.0
1 Results from the GOESClient:
     Start Time           End Time       Source Instrument Wavelength
        str19                str19         str4     str4       str3
------------------- ------------------- ------ ---------- ----------
2011-06-07 06:20:00 2011-06-07 06:30:00   nasa       goes        nan
```

TimeSeries – an object for representing time-domain data

The TimeSeries object is used to represent columns of time-ordered scalar values. Instantiate the TimeSeries object by passing in a file and has convenient plotting methods, e.g.:

```
from sunpy import timeseries
goes = timeseries.TimeSeries('go1520110607.fits')
goes.peek()
```

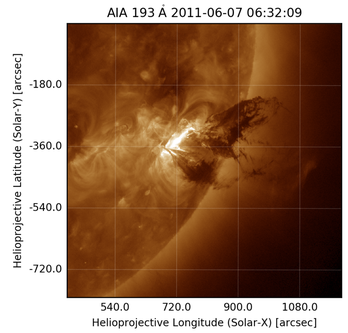


Map – an object for representing image data

The Map object in SunPy provides a convenient framework for manipulating solar images from many sources, e.g.:

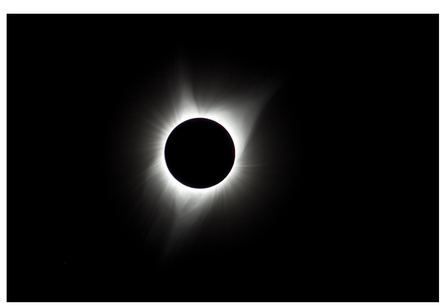
```
import sunpy.map
from astropy import units as u
from astropy.coordinates import SkyCoord
```

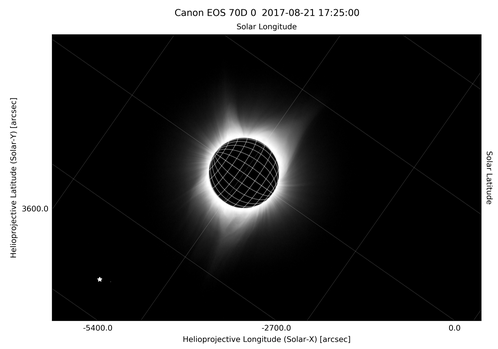
```
aia_map = sunpy.map.Map('AIA20110607_063209_0193.fits')
bottom_left = SkyCoord(400*u.arcsec, -800*u.arcsec)
top_right = SkyCoord(1200*u.arcsec, 0*u.arcsec)
smap = aia_map.submap(bottom_left, top_right)
smap.plot(clim=[0,6000])
```



Use everything together to do cool stuff – like fitting your eclipse photos!

On the left is a photo of this summer's total solar eclipse, as seen from a location about 60 miles north of Boise, Idaho, on August 21, 2017. This image was taken with a Canon EOS 70D by Steven Christe. On the right is the same photo, but fit to a helioprojective coordinate system via a SunPy Map object. This allows us to identify locations on the solar disk as well as other objects. For example, Regulus is indicated by the white star on the bottom left. For the entire code, see: **https://github.com/ehsteve/solar-eclipse**.





Coordinates – an object for representing coordinates

The SunPy coordinates package describes locations in physical space and coordinate frames. It provides a way to transform coordinates from one frame to another. Here we define an observer location using the SkyCoord object:

```
from astropy import units as u
from astropy.coordinates import SkyCoord
from sunpy.coordinates import frames
coord = SkyCoord(-20*u.degree, -56*u.degree, frame=frames.HeliographicStonyhurst)
print(coord)
```

```
<SkyCoord (HeliographicStonyhurst: obstime=None): (lon, lat, radius) in (deg, deg, km)
    (-20., -56.,  695508.)>
```

Citing SunPy

The continued growth and development of SunPy depends on community adoption and awareness. If you use SunPy in your work, we kindly ask that you acknowledge this with the following statement: *"This research has made use of SunPy, an open-source and free community-developed solar data analysis package written in Python (citation)."* The appropriate citation is as follows: The SunPy Community et al., Computational Science & Discovery, Volume 8, No. 1, 2015.