sunpy 0.8: Python for Solar Physics

Andrew R. Inglis[1], Monica Bobra[2], Steven Christe[1], Russell Hewett, Jack Ireland[1], Stuart Mumford[3], Juan Carlos Martinez Oliveros[4], David Perez-Suarez[5], Kevin Reardon[6], Sabrina Savage[7], Albert Shih[1], Daniel Ryan[1], Brigitta Sipocz[8], Nabil Freij[9], on behalf of the SunPy Community

1. NASA Goddard Space Flight Center, USA, 2. Stanford University, USA, 3. Univerity of Sheffield, UK, 4. University of California, Berkeley, USA, 5. Mullard Space Science Laboratory, UK, 6. National Solar Obervatory, USA, 7. NASA Marshall Space Flight Center, USA, 8. University of Hertfordshire, UK, 9. Universitat de les Illes Balears, Spain

SunPy 0.8 – what's new?

SunPy 0.8 has just been released. 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 160+ issues. New headline features with 0.8 include:

- **The TimeSeries object** – this new object handles temporal data products from a wide range of solar instruments. It replaces the LightCurve object, which is now deprecated.
- **New coordinates package** – can now define coordinates and convert easily between solar and astrophysical coordinate systems.
- **The Unified Downloader (FIDO)** – provides a single consistent interface for searching for and downloading data from any supported source.

Scientific Python

Python is a general-purpose, open source and free programming language that is well-suited to scientific investigation. There is a large and growing ecosystem of scientific packages already in place, including:

NumPy – fast array operations
SciPy – common scientific operations, e.g. fitting
Pandas – time series manipulation
AstroPy – astrophysical tools
Scikit-image – image processing
Scikit-learn – machine learning

This provides an ideal environment to develop solar tools in Python.

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)."

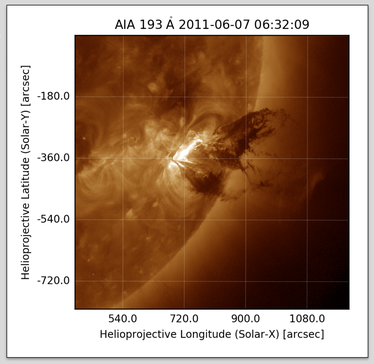
The citation refers to the most recent SunPy paper:

The SunPy Community et al., Computational Science & Discovery, Volume 8, No. 1, 2015

Map - an object for representing and working with image data

The Map object in SunPy provides a convenient framework for manipulating solar images from many sources, for example from SDO/AIA. Maps are coordinate-system and unit aware, and thus can be easily converted to different coordinate systems or overlayed.

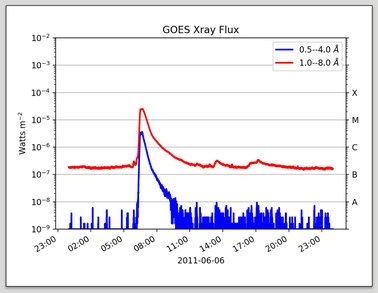
```
import sunpy.map
from astropy import units as u
from astropy.coordinates import SkyCoord

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])
```



TimeSeries – an object for working with time-domain data

The TimeSeries object is new in SunPy 0.8, and replaces the LightCurve object. The downloading of data is no longer done with TimeSeries – instead, the object operates on existing files. TimeSeries supports every data source that was previously supported by LightCurve.

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



Coordinates – new tools for representing and converting coordinate frames

```
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.)>
```

The Unified Downloader (FIDO) – a tool to search for and download data

Using the Unified Downloader module it is now easy to download data from multiple solar sources simultaneously using SunPy. Right shows an example of how complex queries can be constructed to retrieve images and timeseries data from different instruments at the same time. FIDO knows to query multiple clients to find the data it needs.

The data can easily be downloaded using the `Fido.fetch()` command.

```
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
```

SunPy on the web

sunpy.org - The SunPy website.
https://github.com/sunpy/sunpy - the SunPy codebase.
docs.sunpy.org – the documentation for all the SunPy code.
http://docs.sunpy.org/en/stable/generated/gallery/index.html - the SunPy gallery, contains lots of useful examples of what SunPy can do.
sunpy@googlegroups.com - the general SunPy mailing list.
sunpy-dev@googlegroups.com - mailing list for technical inquiries.
https://riot.im/app/#/room/#sunpy-general:matrix.org - Matrix channel

Visit our website:

Contribute

Provide feedback

We could always use more voices and opinions in the discussions about SunPy and its development. You may want to suggest a new feature or a change to existing code. There are a number of ways to make your voice heard and we actively want feedback from the community.

Report bugs

If you run into unexpected behavior or run into a bug please report it. All bugs are kept track of on our issue tracker on GIthub:
https://github.com/sunpy/sunpy/issues

Provide code

If you want to contribute software to the SunPy repository, you are very welcome, but we do encourage you to communicate and work with existing developers, as all code undergoes detailed review and testing before it is accepted into SunPy, even modifications by the lead developer. SunPy and both use the very powerful git distributed version control software to track changes to the code and manage branches and releases. Anyone is free to 'fork' the SunPy project on Github, modify it and contribute those changes back to SunPy via a 'pull request'.

Acknowledgements

Many of the features in SunPy are the result of generous contributions from external organisations. In particular, SunPy would like to thank the Google Summer of Code (GSOC) and the European Summer of Code in Space (SOCIS) programmes, which have provided funds for summer students to work on SunPy development for several years. Since 2011, over a dozen summer students have helped develop key SunPy features via these programmes.