AAS/SPD 2014 218.39

SunPy The community-developed, free and open-source solar data analysis environment for Python.

Steven Christe[1], Stuart Mumford[2], David Perez-Suarez[3], Jack Ireland[4], Albert Y. Shih[1], Andrew Inglis[5], Simon Liedtke[6], Russel Hewett[7], the SunPy Collaboration
[1]Solar Physics Lab, NASA GSFC, Greenbelt, MD, US. [2]Solar Physics and Space Plasma Research Centre, University of Sheffield, Sheffield, UK.
[3]South African National Space Agency - Space Science, Western Cape, South Africa. [4]ADNET systems, Greenbelt, MD, US.
[5]Catholic University, Washington , DC, US. [6]University of Bremen, Bremen, GE. [7]Department of mathematics, MIT, Cambridge, MA, US

Overview

SunPy is a free and open source Python library which provides tools for the acquisition, processing and visualisation of solar data. Over the three years of SunPy's development, the code base has grown to over 17,000 lines. SunPy is already a useful package for the analysis of calibrated solar data, and it continues to gain significant new capabilities with each successive release. The primary focus of the SunPy library is the analysis and visualisation of 'high-level' solar data. This means data that has been put through instrument processing and calibration routines, and contains full (WCS) coordinate information. The plan for SunPy is to continue development within this scope. The primary components of this plan are to provide a set of data types (Map, MapCube, CompositeMap, Lightcurve, Spectra) that are interchangeable with one another: e.g., if you slice a MapCube along one spatial location, a LightCurve of intensity along the time range of the MapCube should be returned. In concert with the work on the data types, further integration with the astropy package will enable SunPy to incorporate many new features.

Python

Python is an open source, free, and general purpose programming language that is widely used for a variety of applications. It also has a thriving ecosystem of scientific-oriented packages which provide things like high-speed array operations (Numpy), common scientific operations (SciPy), publication ready visualisation (matplotlib) as well as symbolic algebra (SymPy) and more focused packages such as scikit-image for image processing and Astropy a core package for astronomy.

Python is well-suited to scientific investigation.

- Its emphasis on readability encourages the creation of reusable code. This in turn leads to easier collaboration between researchers. Additionally, documentation is built-in to the language itself.
- Python provides a good mixture of high level and low level programming. Basic Python is high-level but low-level/fast code is also possible through Python's great support for other languages (e.g. C/C++, FORTRAN).
- Python also provides a good testing framework which enables code to be more stable over long-term use.

SunPy 0.5

The next version of SunPy (0.5) will soon be released. Major new features include

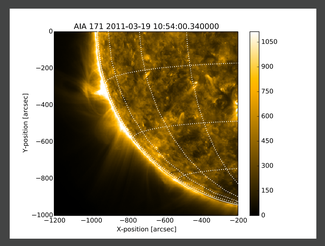
- Added Nobeyama Radioheliograph lightcurves
- Upgrade to existing GOES lightcurves, now provides full metadata and data back to 1981.
- Added support for NOAA solar cycle measurements for both past and predictions.
- Added automatic image coalignment with support for MapCubes.
- Major bug fixes and updates to automatic testing (which should mean fewer bugs in the future!).
- Community "steering committee" created and formal process for enhancing SunPy defined.
- For more see github.com/sunpy/sunpy/blob/master/CHANGELOG.md

Examples

Map

The map is created from an SDO/AIA FITS file, a cutout of the map is created, and then a quick-view plot is created with lines of heliographic longitude and latitude over-plotted.

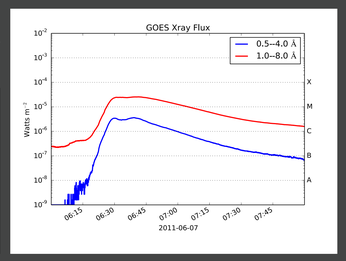
```
>>> import sunpy.map
>>> aiamap = sunpy.map.Map('aia_file.fits')
>>> smap = aiamap.submap([-1200, -200], [-1000, -0])
>>> smap.peek(draw_grid=True)
```



Lightcurves

Example retrieval of a GOES lightcurve using a time range and the output of the peek() method. The maximum flux value in the GOES 1.0–8.0 A channel is then retrieved along with the location in time of the maximum.

```
>>> from sunpy import lightcurve
>>> goes = lightcurve.GOESLightCurve.create(
          '2011-06-07 06:00','2011-06-07 08:00')
>>> goes.peek()
>>> print('The max flux is ' +
          str(goes.data['xrsb'].max()) + '
          at ' + str(goes.data['xrsb'].idxmax()))
The max flux is 2.5554e-05 at 2011-06-07 06:41:24
```



Solar Data Search and Retrieval

SunPy provides well-developed interfaces to both the VSO and HEK which provides advanced searches capabilities not available elsewhere using logical operators.

```
>>> from sunpy.net import vso
>>> client = vso.VSOClient()
>>> tstart, tend = '2011/6/7 05:30', '2011/6/7 10:30'
>>> condition = (vso.attrs.Detector('cor1') | vso.attrs.Wave(125, 135) |
                 vso.attrs.Wave(165, 175) )
>>> advanced = client.query(vso.attrs.Time(tstart, tend), condition)
>>> results = client.get(advanced)
```

```
>>> from sunpy.net import hek
>>> client = hek.HEKClient()
>>> tstart, tend = '2011/08/09 00:00:00', '2011/08/10 00:00:00'
>>> result = client.query(hek.attrs.Time(tstart, tend), hek.attrs.EventType('FL'),
                         (hek.attrs.Event.Coord1>50) or
                         (hek.attrs.FL.PeakFlux>1000.0))
>>> len(result)
48
```

Solar Constants

The sun.constants module provides a number of solar-related constants in order to provide consistency in the calculations of derived solar values within the SunPy code base, but also to the user.

```
>>> from sunpy.sun import constants
>>> print(constants.mass)
  Name   = Solar mass
  Value  = 1.9891e+30
  Error  = 5e+25
  Units  = kg
  Reference = Allen's Astrophysical Quantities 4th Ed.
# Calculate the average density of the Sun and convert to cgs
>>> (constants.mass/constants.volume).cgs
<Quantity 1.40851154227 g / (cm3)>
```

Contribute

Provide Feedback
We could always use more voices and opinions in the discussions about SunPy and its development from both users and developers. You may want to suggest a new feature or describe how something is not working how you think it should. There are a number of ways to make your voice heard and we would love to hear from you (see below).

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. You can add a bug report there or if you are not sure it's a bug send an email to the developer mailing list.

Provide Code
SunPy library is developed in the open on GitHub, where anyone is free to 'fork' the project, modify it and contribute those changes back to SunPy via a 'pull request'. SunPy aims to foster a friendly and collaborative atmosphere where all contributions no matter their size are welcome.
This workflow enables very detailed review and testing of code before it is accepted into SunPy. SunPy and both use the very powerful git distributed version control software to track changes to the code and manage branches and releases.

Resources

sunpy.org - the website (where you can find all the following links)
docs.sunpy.org - the documentation
sunpy@googlegroups.com - the email list
sunpy-dev@googlegroups.com - the developer mailing list
webchat.freenode.net/?channels=sunpy - the chat room
github.com/sunpy/sunpy - the code
github.com/sunpy/sunpy/issues - the bug tracker
sunpy.org/blog/index.html - the blog
sunpy.org/2014/04/03/RHESSI-Workshop-13 - quick start tutorials

Feedback & Acknowledgements

Any comments or suggestions are appreciated.

steven.christe@nasa.gov

Many of the features in SunPy have been (and are currently being) developed with the generous support of external organizations. Initial development of SunPy's VSO and HEK implementations were funded by ESA's Summer of Code In Space (SOCIS 2011, 2012, 2013) program. In 2013, with support from Google's Summer Of Code (GSOC) program, through the Python Software Foundation, the helio, hek2vso, and database subpackages were developed. The Spectra and Spectrogram classes were implemented with support from the Astrophysics Research Group at Trinity College Dublin, Ireland, in 2012. SunPy is currently funded by GSOC 2014.