



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

Laura A. Hayes¹
on behalf of The SunPy Community
22 April 2020 PyAstro

¹NASA Goddard Space Flight Center/USRA



Solar Physics

What do solar physicists do? What do we need in software?

- **Study of the Sun**

- *solar events (flares, coronal mass ejection), Sun-as-a-star, space weather, ‘plasma lab in the sky’*

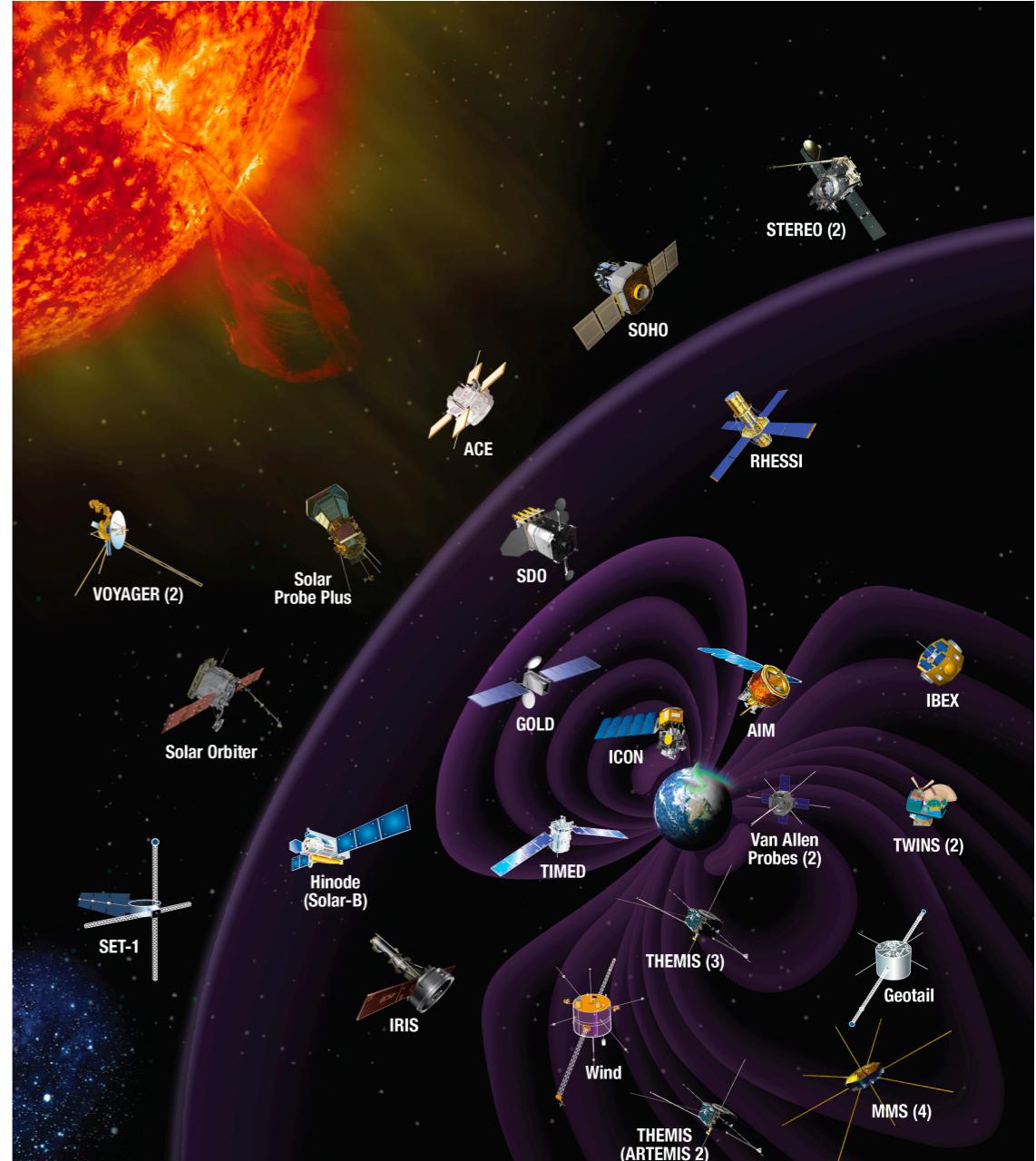
- **Detailed measurements**

- From both *space- and ground- based instruments* - lots of data!
- Images, time-series, spectra

- **What software do we need?**

- Download data, read in data into standard format, deal with different coordinates systems and transformations

- Data exploration/visualisation e.t.c



Heliophysics system observatory credit NASA



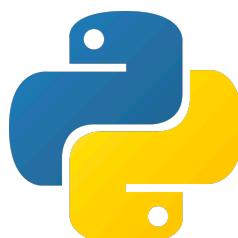
Software in Solar Physics

Whats currently available?



SolarSoftWare (SSW) *Interactive Data Language (IDL)*

- 2+ decades of instrument specific software
- Large amount of software (4 million lines code! 16GB)
- Open sourced, but closed development
- Not version controlled, not coordinated/organized, conflicts



SunPy *Python*

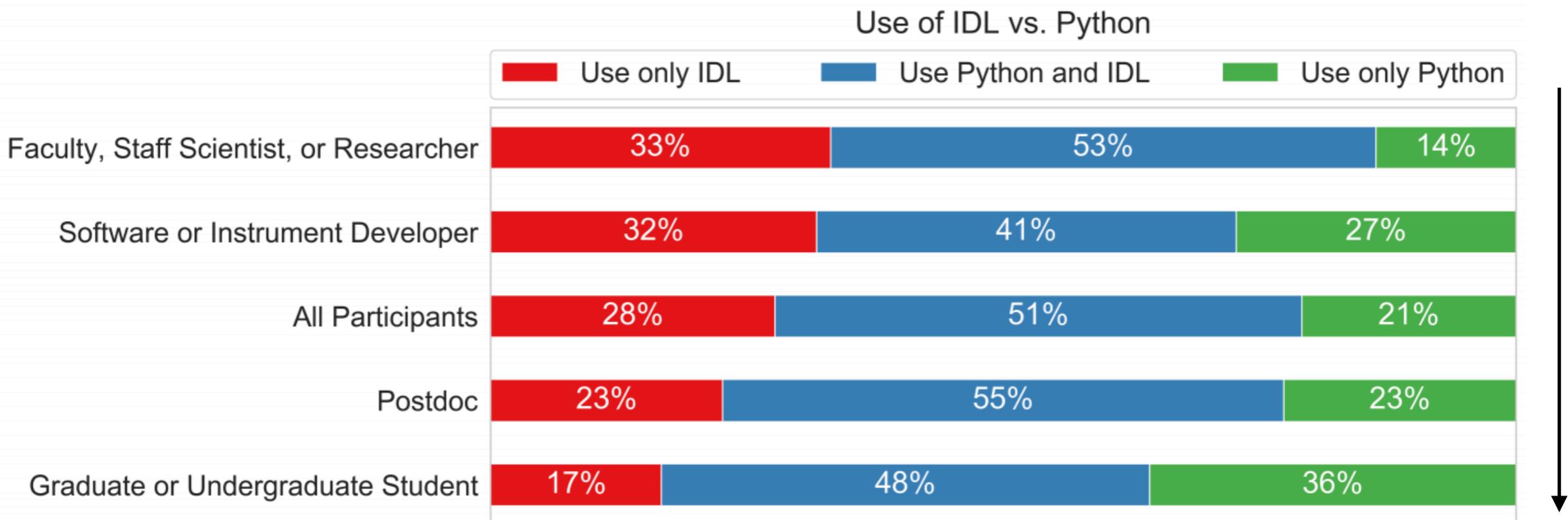
- More to come in this talk ;)



Software in Solar Physics

Community Survey

- 13-question survey to understand the software and hardware usage of the solar physics community (364 people, across 35 countries took part)
- Compared to astrophysics - **IDL is still very common**, especially with faculty, staff researchers etc.



Bobra, Monica G., et al. "A Survey of Computational Tools in Solar Physics." arXiv preprint arXiv:2003.14186 (2020).



SunPy

What is SunPy?

Python in
Heliophysics (PyHC)

The SunPy project facilitates and promotes the use and development of several community-led, free, and open source data analysis software packages for solar physics based on the scientific Python environment.

Functionality

- Provide Python tools specific to solar data analysis
- Focus on calibrated high level data
- Provide standardized gateway for solar data into the scientific python environment
- Support other solar packages outside the scope of SunPy core

Cultural

- Open-source community and inclusive of everyone (anyone can contribute!)
- Coordinate development
- Code testing and code review
- Version control
- Standardized and discoverable documentation

SunPy

What is SunPy?



Python in
Heliophysics (PyHC)



The SunPy project facilitates and promotes the use and development of several community-led, free, and open source data analysis software packages for solar physics based on the scientific Python environment.

Functionality

- Provide Python tools specific to solar data analysis - gateway into ecosystem
- Focus on calibrated high level data
- Leverage mature and maintained code from other field (e.g. astropy) 
- Support other solar packages (affiliated) outside the scope of SunPy core

Cultural

- Open-source community and inclusive of everyone (anyone can contribute!)
- Coordinate development
- Code testing and code review
- Version control
- Standardized and discoverable documentation



SunPy Organization

The SunPy Community

SunPy Board

Up-to 10 people, guides overall direction of sunpy project by voting on SunPy Enhancement Proposals (SEPs) - anyone can propose

Lead-Developer

Leads development core team and community developers and affiliated packages

Deputy Lead-Developer

Assists lead developer

Community Roles

Sub-package maintainers

Assist lead devs with maintaining sub-packages

Summer of Code students

Users

Community Developers

<https://sunpy.org/project/>



Contributing to SunPy

What does this look like?

Anyone can contribute! ☀️

Guidelines in place to ensure new code meets quality standard:

Lots of assistance
given to
newcomers!

Style Guide

All code and docs must comply with widely used style guides (e.g. Pep8)

Docs

New features require documentation - code comments, docstrings and guide, examples

Unit tests

All code must provide unit tests that cover functionality

Within Scope

All code must be within scope and approved by at least 2 members of dev community

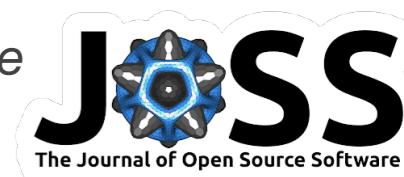
Automated testing on GitHub (incl. different op systems, building docs, plots, code-coverage etc). Use of Azure, CircleCI,Codecov and Travis CI



Where is SunPy now?

Current Status

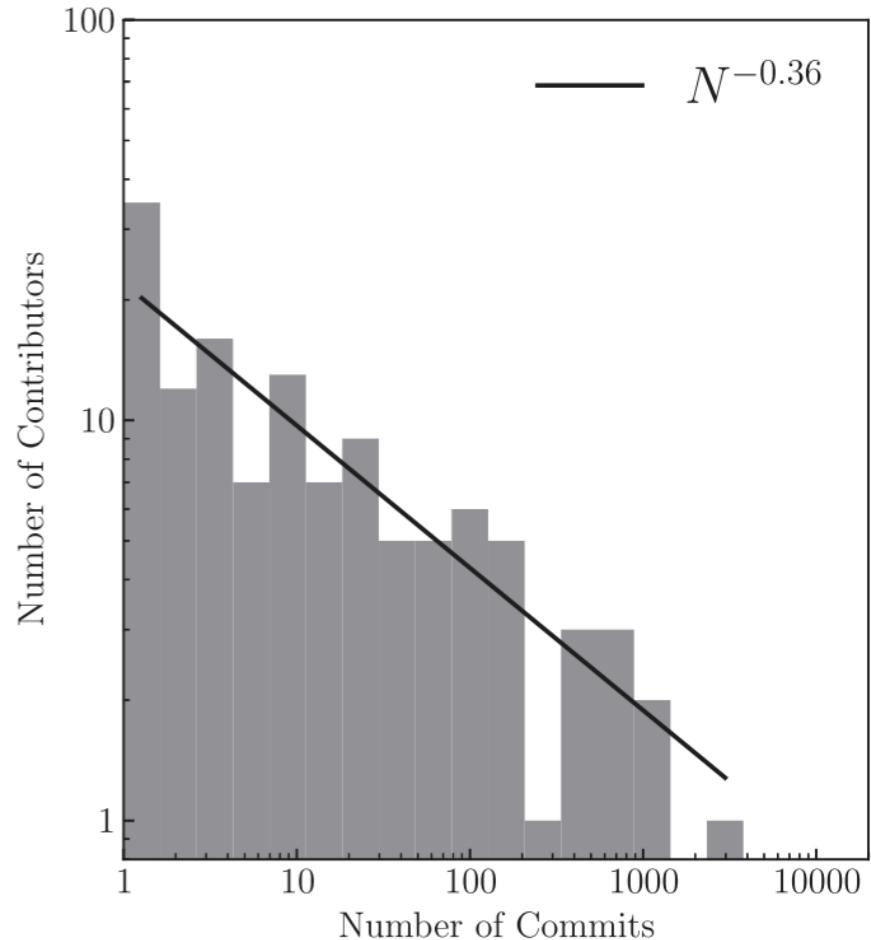
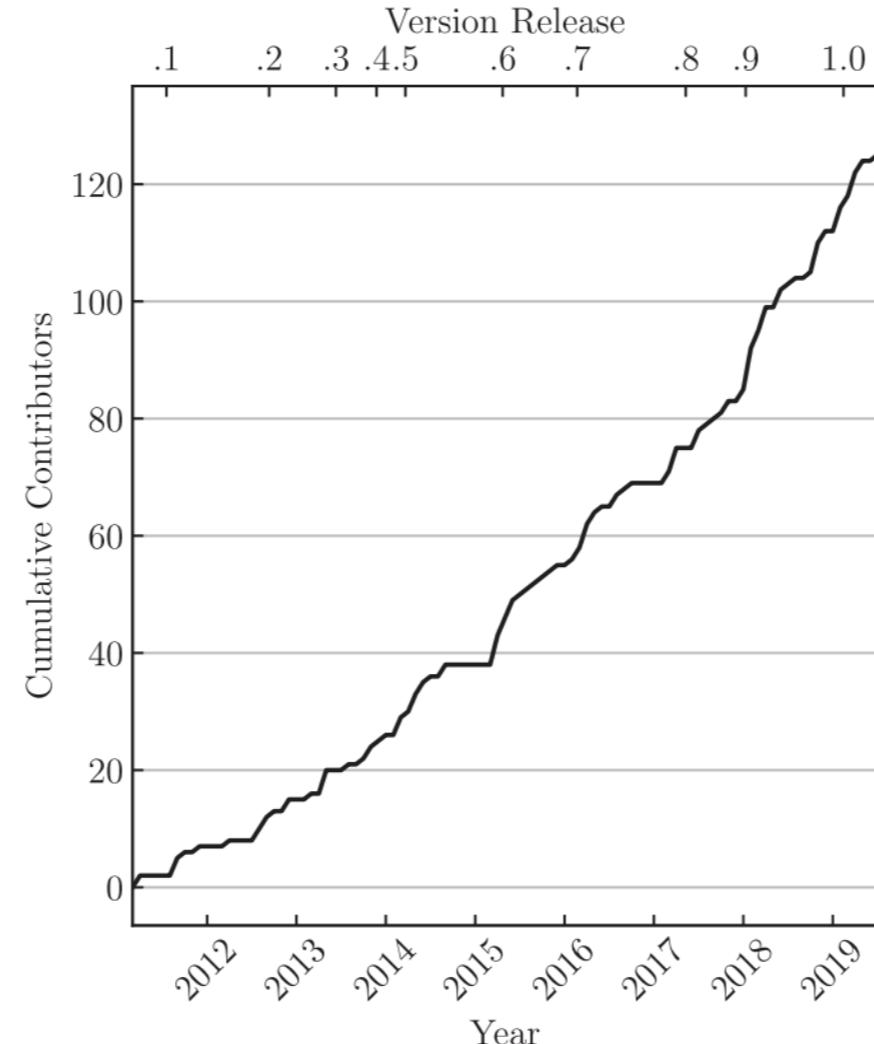
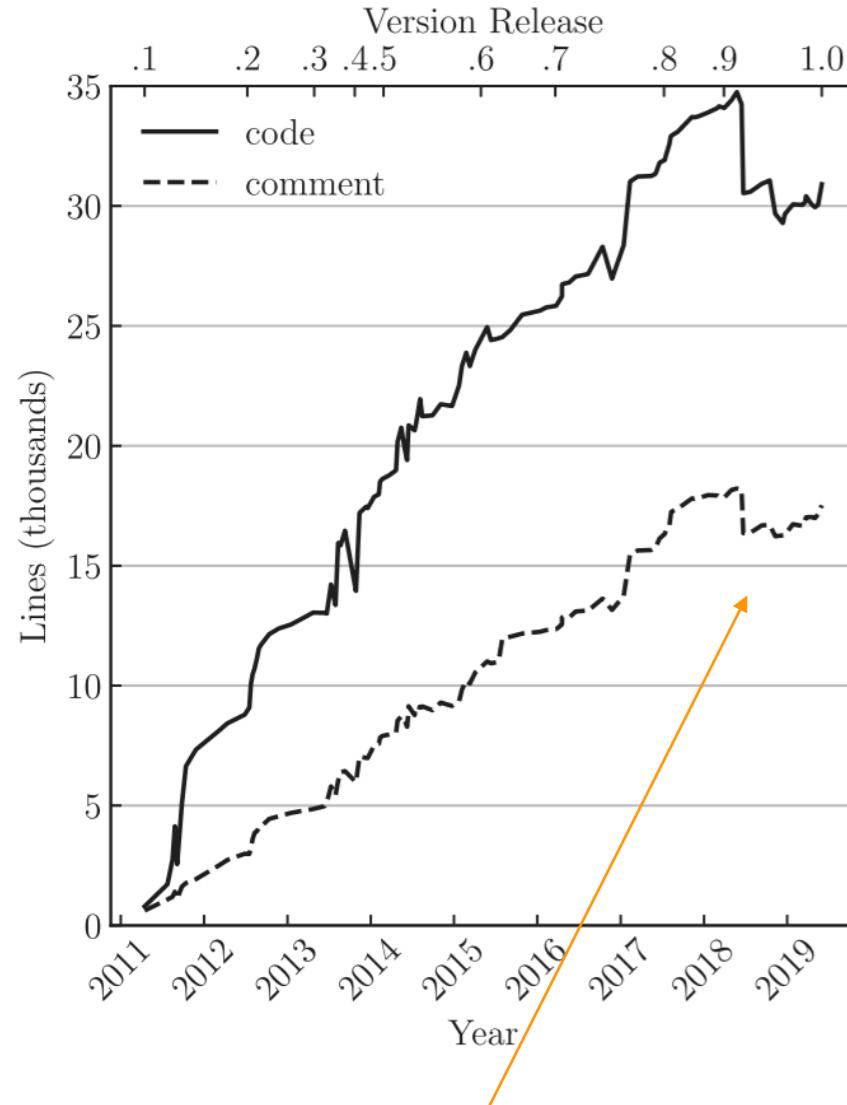
- Project officially founded in March 2014, began in March 2011 (9 years ago!)
- Released SunPy 1.1 (1.0 first stable release)
- Now release schedule - twice a year
- ~50,000 lines of code (incl. comments & docs), 134 unique contributors
- Published paper and code:
 - *The SunPy Community, et al. "The sunpy project: Open source development and status of the version 1.0 core package." The Astrophysical Journal 890.1 (2020): 68.*
 - *Mumford, Stuart, et al. "SunPy: A Python package for Solar Physics." Journal of Open Source Software 5.46 (2020).*





Where is SunPy now?

Current Status



Major tidy up for 1.0

Slope quite steep
hope to flatten this



SunPy 1.0 +

Overview and Highlights

- **Headline core changes:**
 - Major clean up of core
 - Improved download capabilities (`parfive`)
 - Improved Coordinates functionality - Sun-specific transformation stack and coordinate frames
 - Full adoption of astropy time
 - Logging system

Data Retriever

sunpy.net and Fido



- **Fido** Unified API for searching and downloading solar data from various search engines and data sources (e.g. VSO, JSOC, https, ftp)

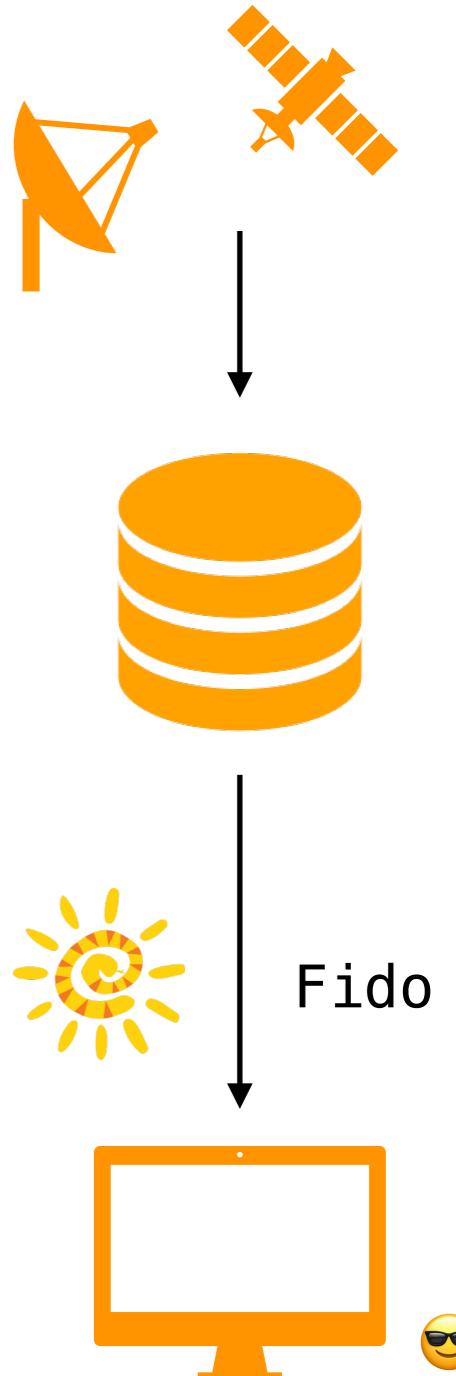
```
In [5]: from sunpy.net import Fido, attrs as a  
  
In [6]: result = Fido.search(a.Time('2012/3/4', '2012/3/6'), a.Instrument('XRS'))  
  
In [7]: result  
  
Out[7]: Results from 1 Provider:
```

3 Results from the XRSCClient:

Table length=3

Start Time	End Time	Source	Instrument	Wavelength
str19	str19	str4	str4	str3
2012-03-04 00:00:00	2012-03-04 23:59:59	nasa	goes	nan
2012-03-05 00:00:00	2012-03-05 23:59:59	nasa	goes	nan
2012-03-06 00:00:00	2012-03-06 23:59:59	nasa	goes	nan

```
In [*]: Fido.fetch(result, path='./{file}')  
  
Files Downloaded: 33%  1/3 [00:00<00:00, 2.90file/s]
```

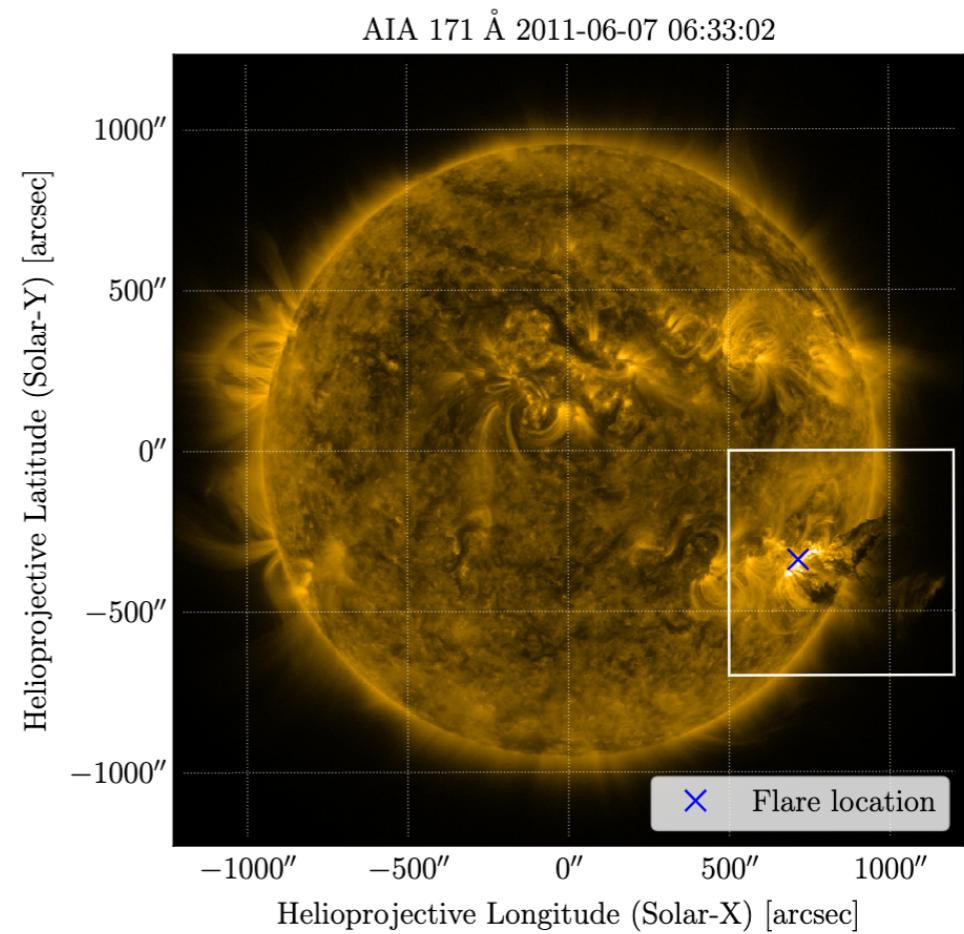
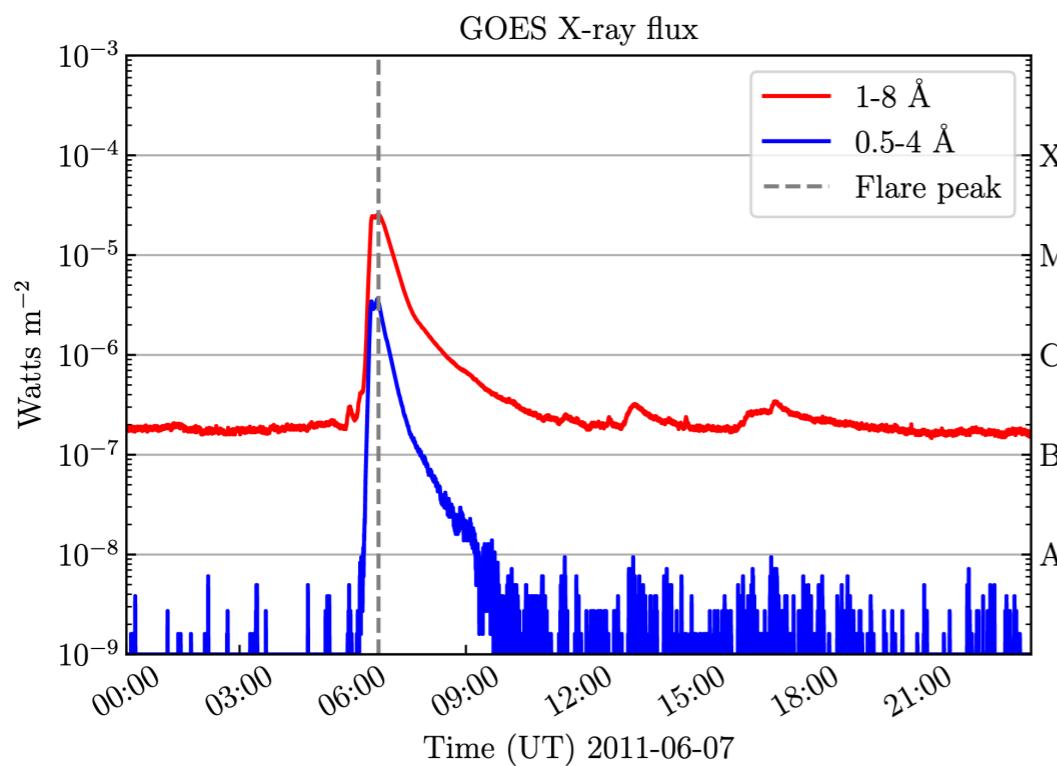


Data Containers

TimeSeries and Map



- SunPy provides general, **standard and consistent interface** for loading and representing solar data across different instruments and missions.



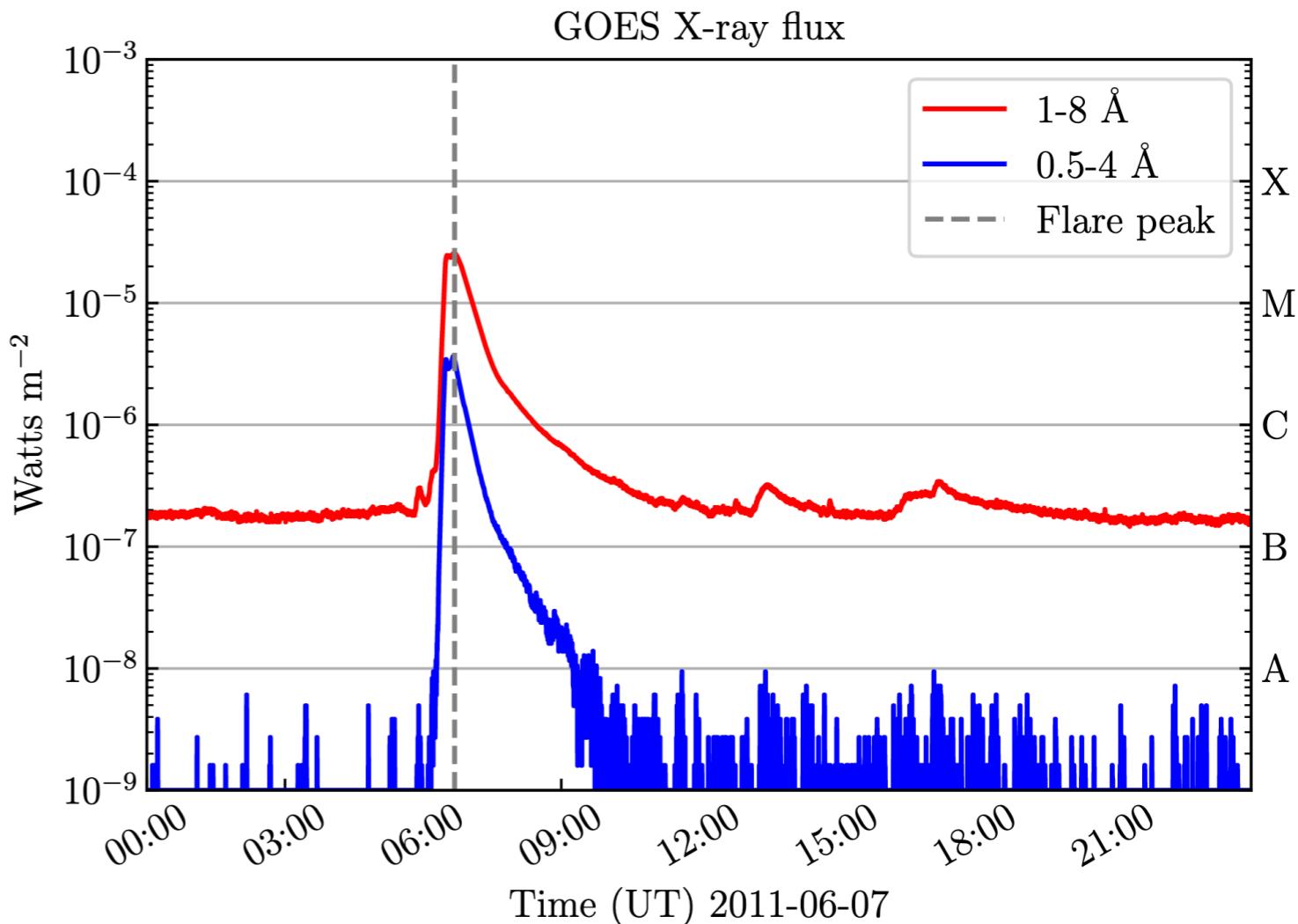
TimeSeries: 1D temporal data

Map: 2D image data

TimeSeries

Data Containers

- MetaData container to hold information from different files, instruments e.t.c
- Identify metadata that corresponds to given times and columns
- Truncate, concatenate, resampling, plot



```
In [2]: from sunpy import timeseries as ts
In [3]: goes_ts = ts.TimeSeries('go1520120304.fits')
In [4]: goes_ts.units
Out[4]: OrderedDict([('xrsa', Unit("W / m2")), ('xrsb', Unit("W / m2"))])
```

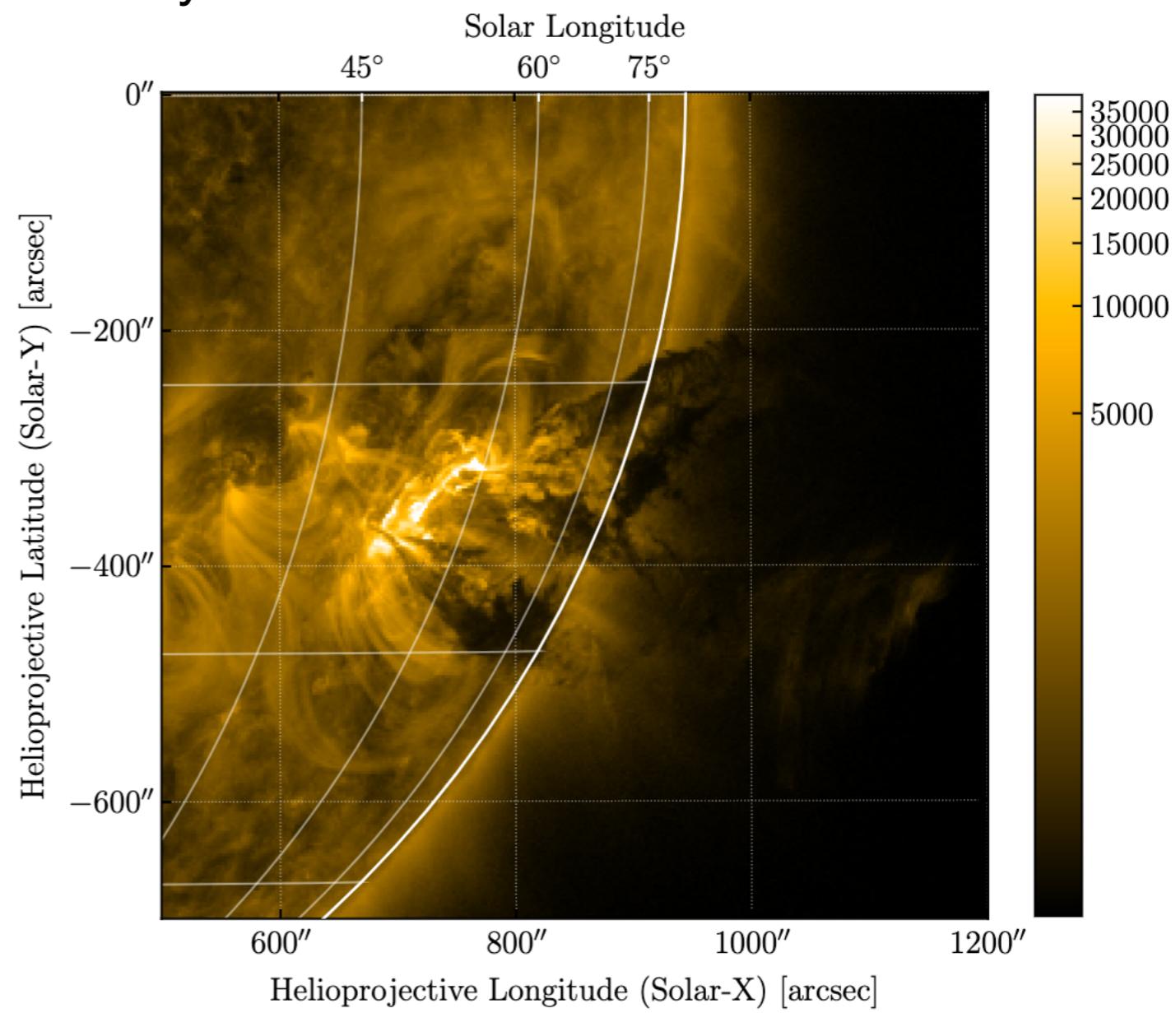
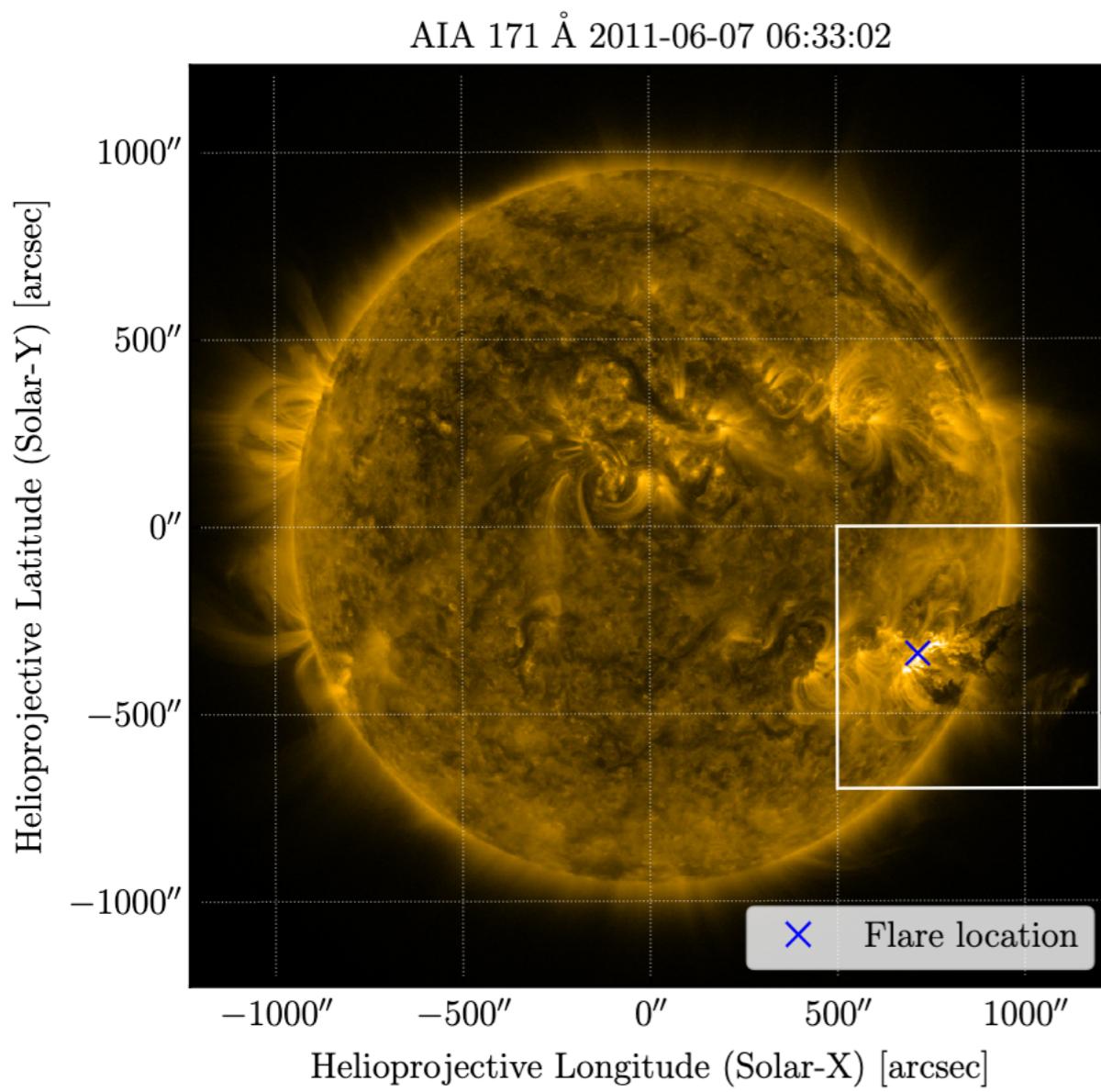
```
In [5]: goes_ts.meta
Out[5]:
```

TimeRange	Columns	Meta
2012-03-03 23:59:58.438999 to 2012-03-04 23:59:56.301999	xrsa xrsb	simple: True bitpix: 8 naxis: 0 extend: True date: 26/06/2012 numext: 3 telescop: GOES 15 instrume: X-ray Detector object: Sun origin: SDAC/GSFC ...



Map Data Containers

- Standardized interface to coordinate aware 2-D images
- Uses WCSAxes for plotting functionality

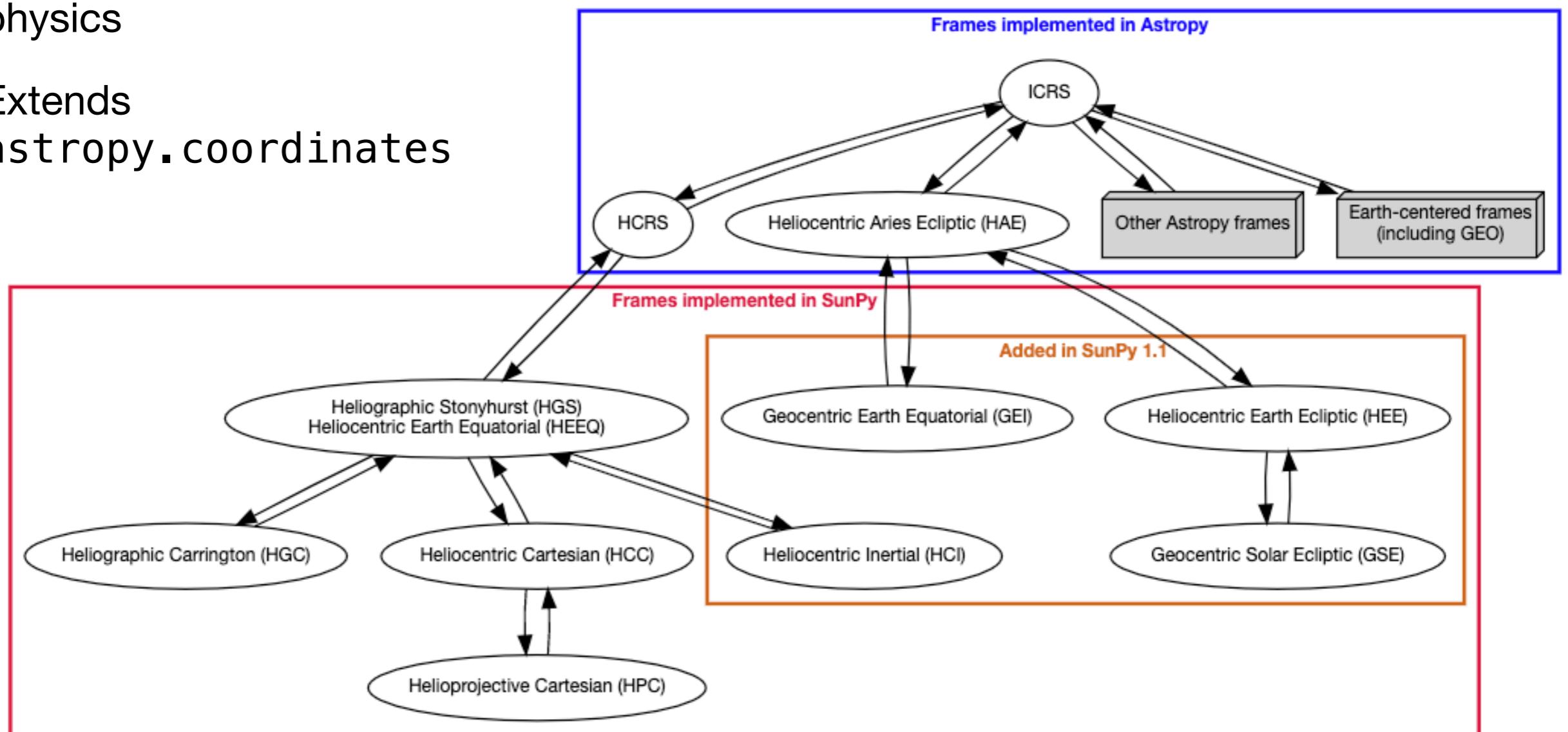




Coordinates

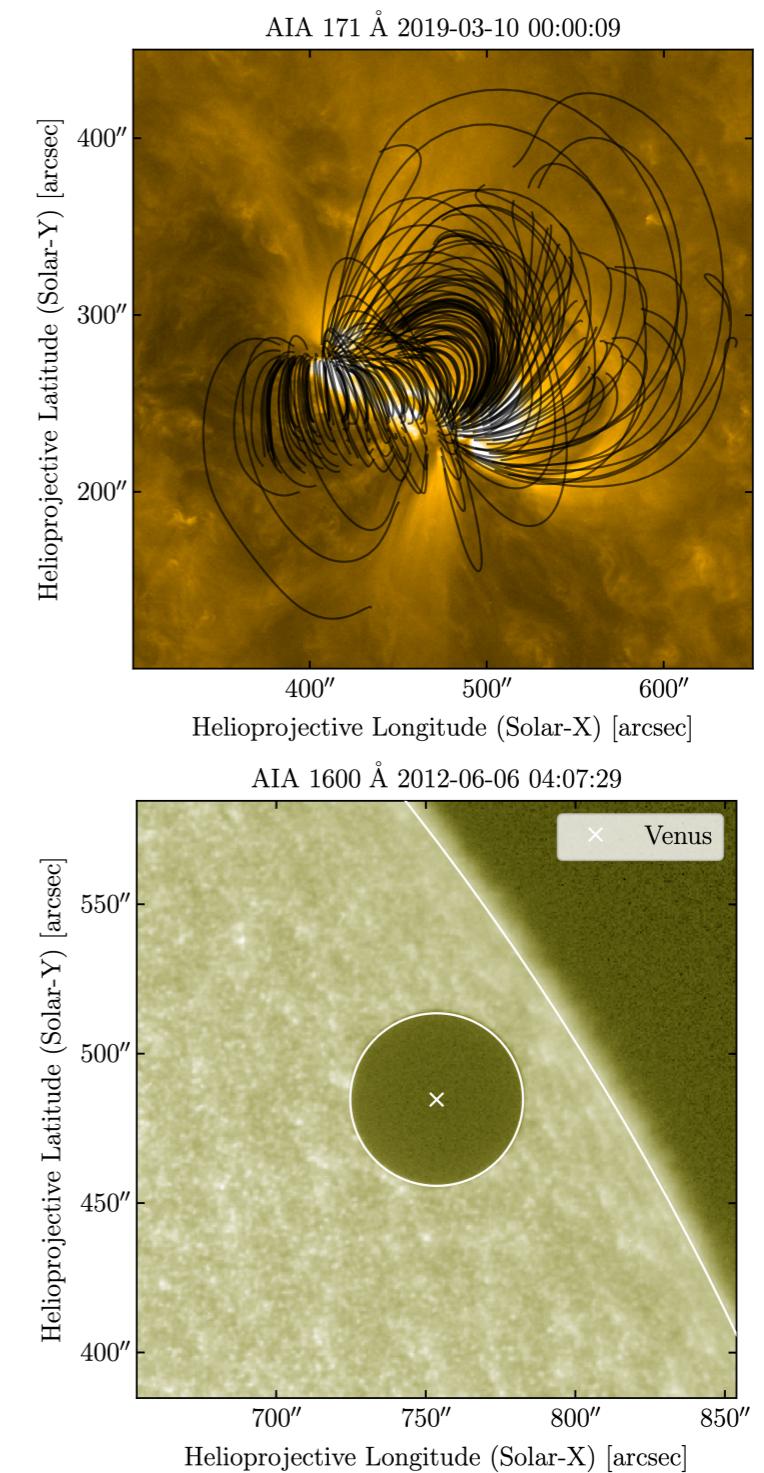
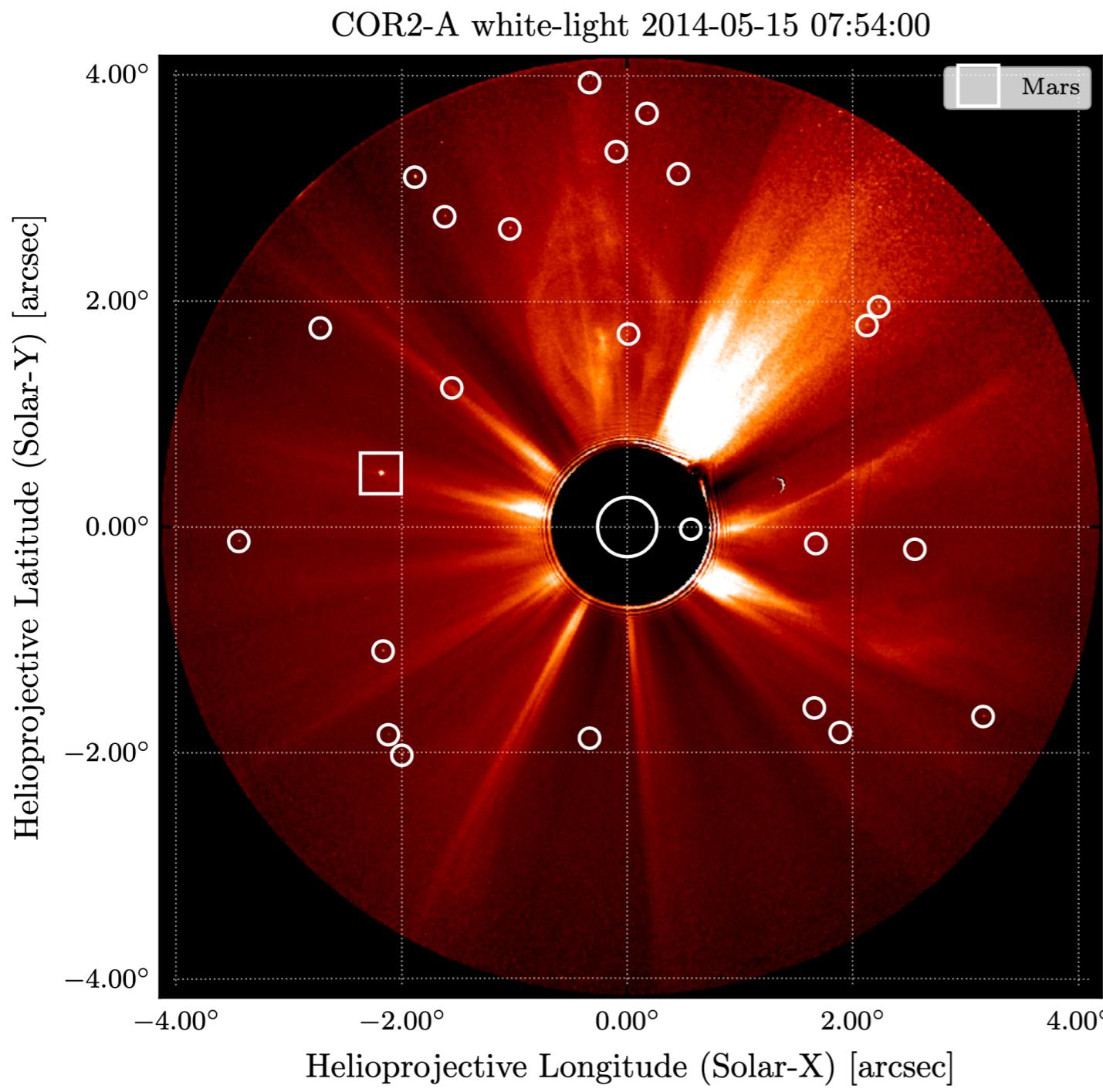
Solar Coordinates

- `sunpy.coordinates` for representing and **transforming** coordinates used in solar physics
- Extends `astropy.coordinates`



Coordinates

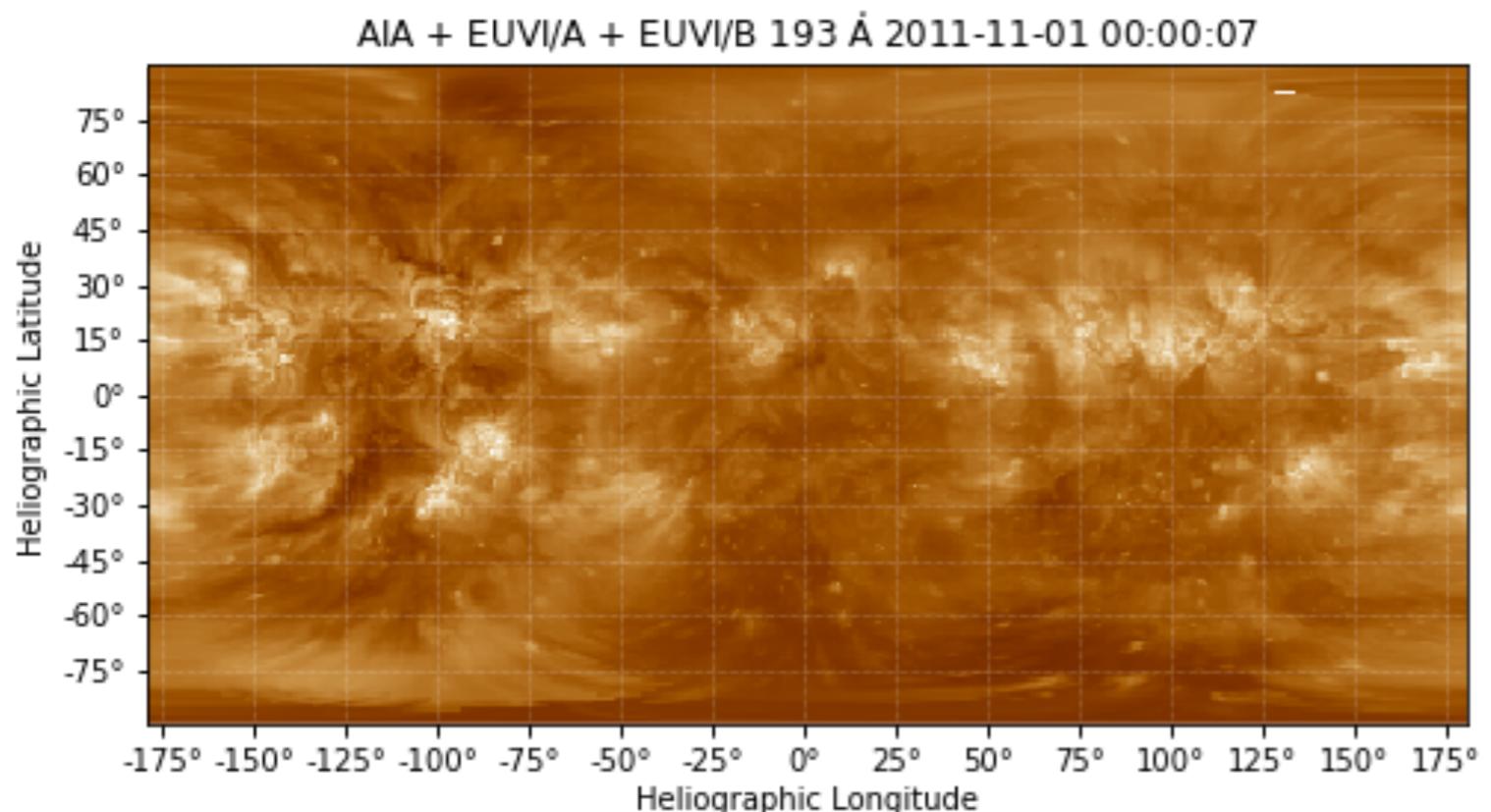
Some Examples





Coordinates

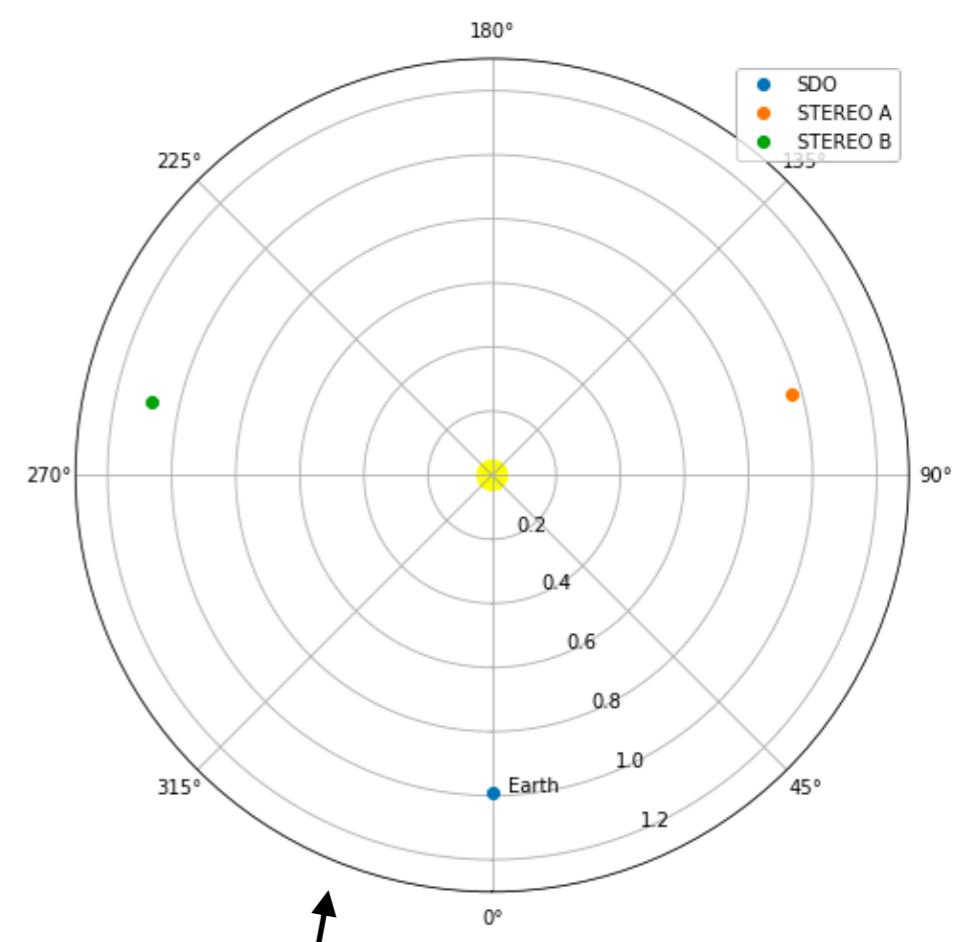
Full map of Sun using different spacecraft



Creation of full Sun map from SDO/AIA
and STEREO A & B

Check this out in our example gallery!
<https://docs.sunpy.org/en/stable/>

using reproject package

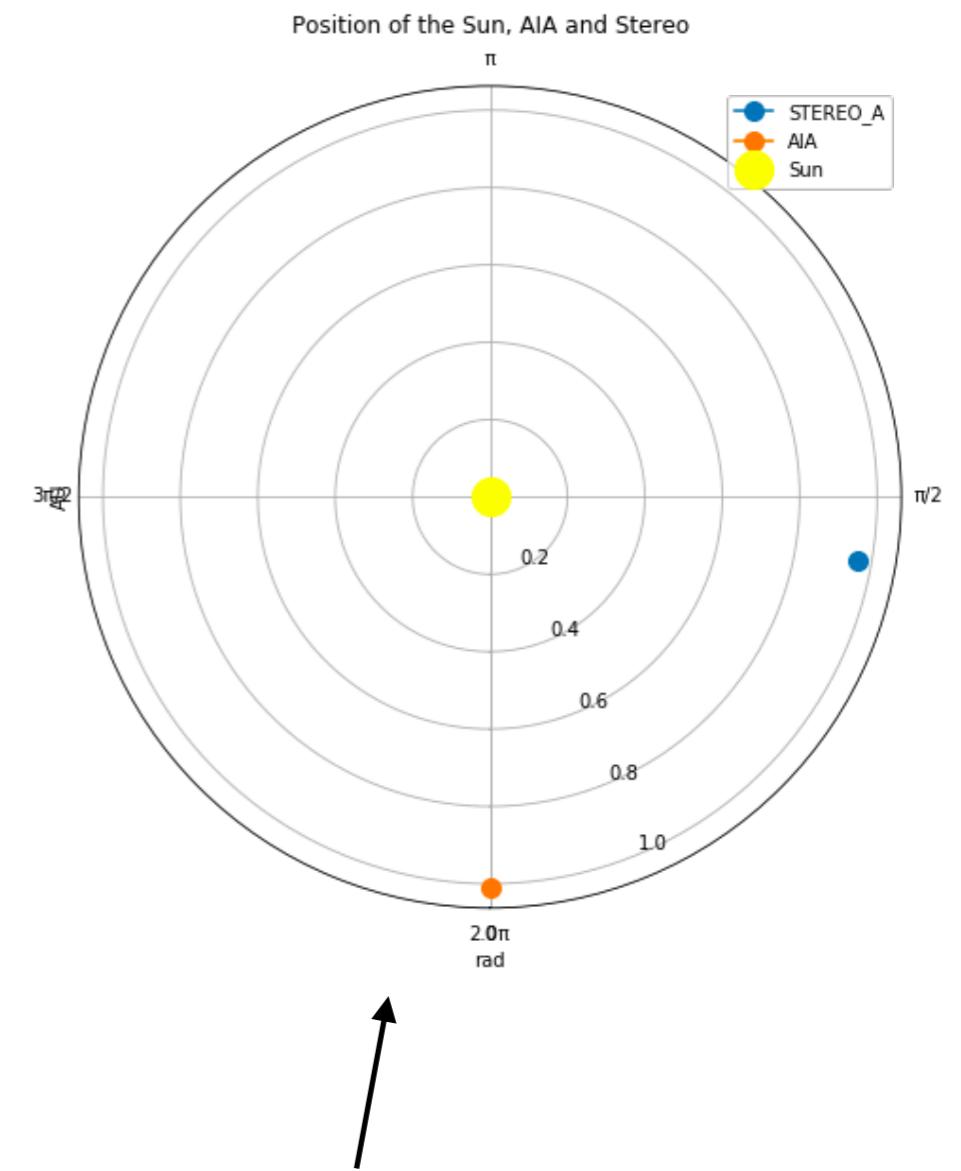
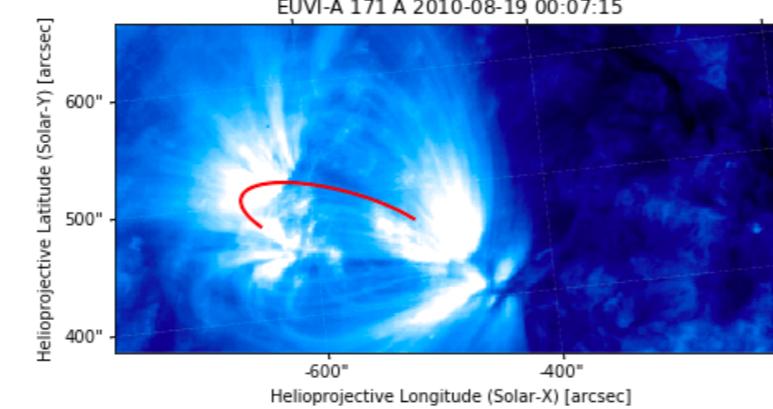
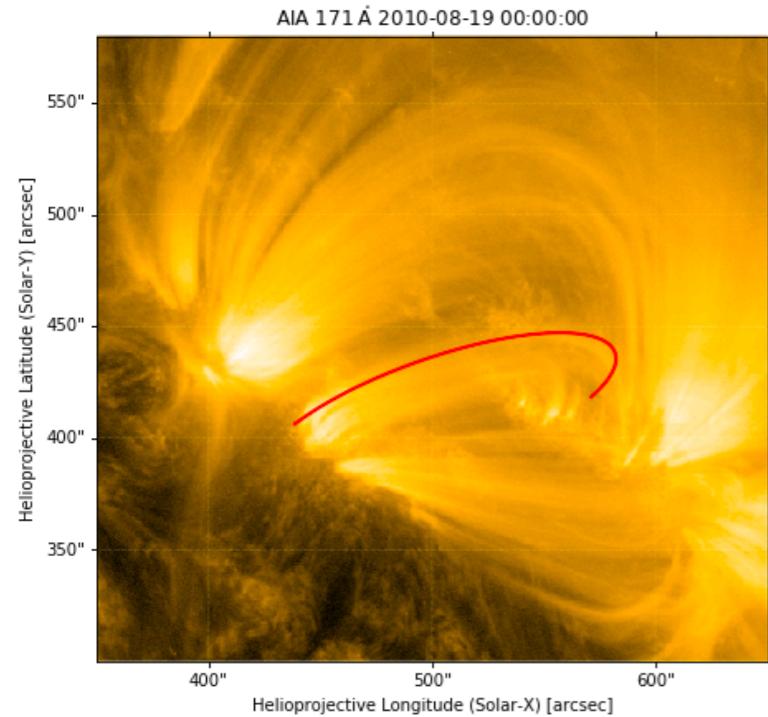
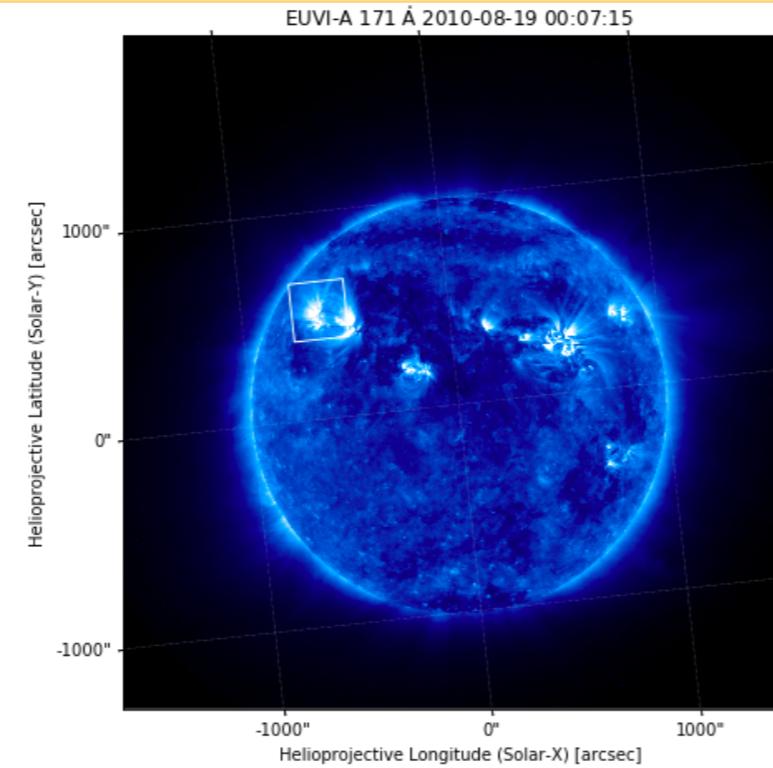
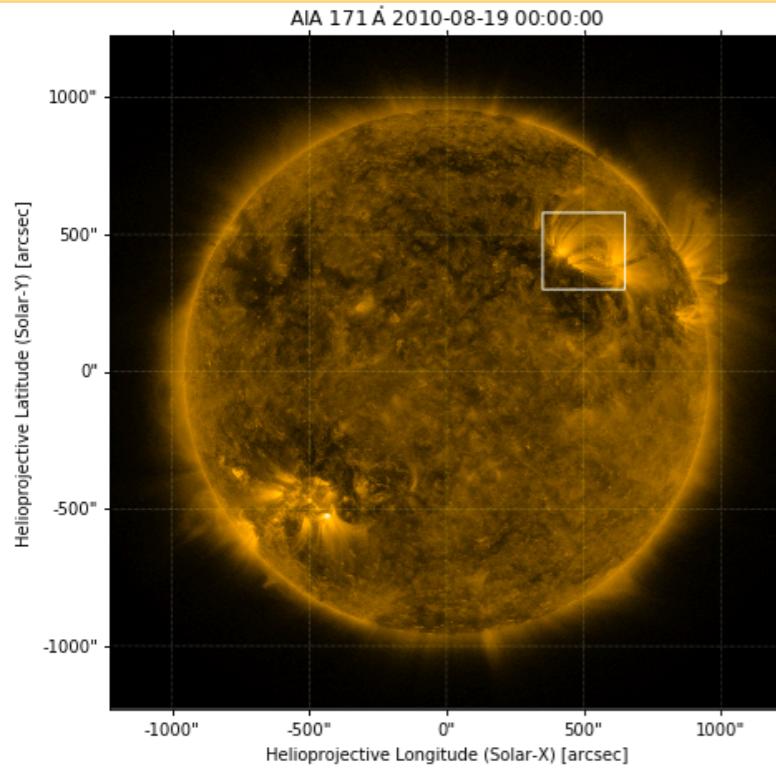


Positions of satellites



Coordinates

Finding regions of interest

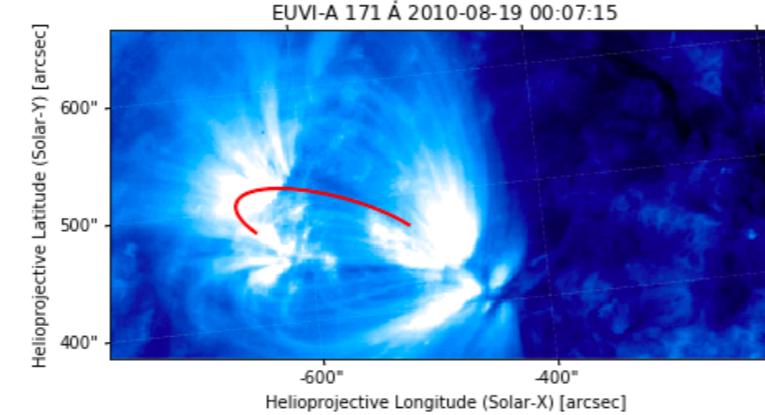
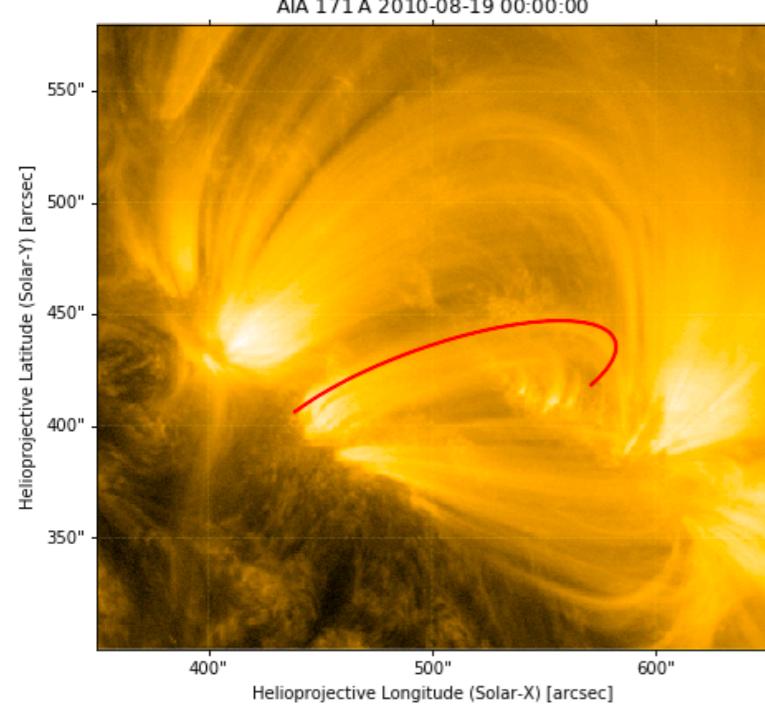
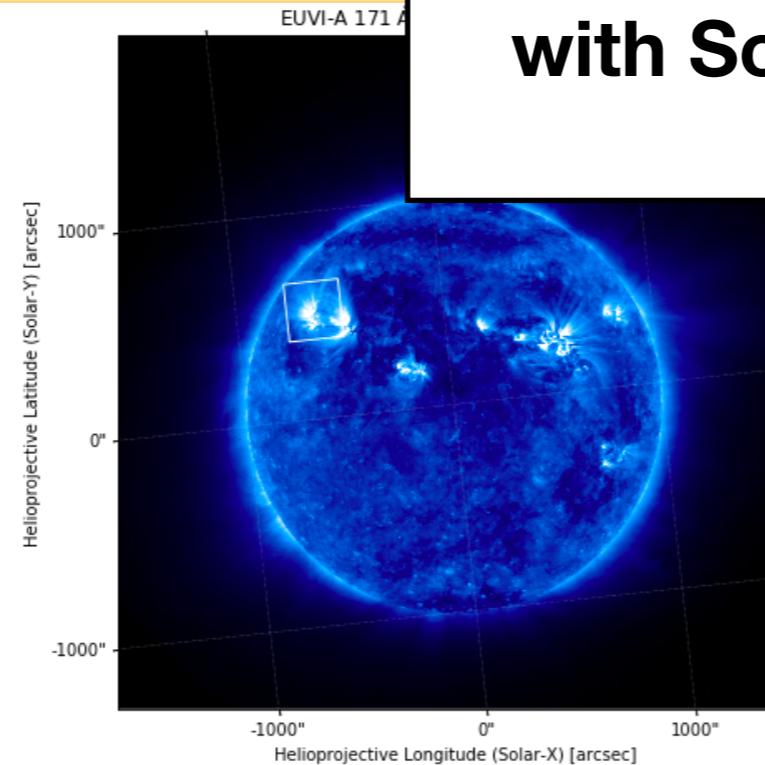
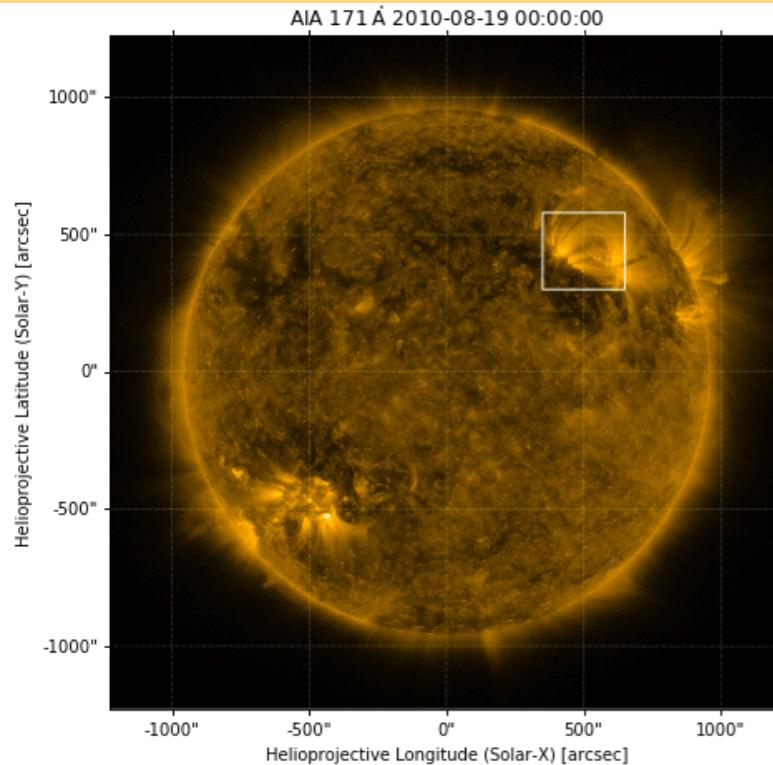


Positions of satellites

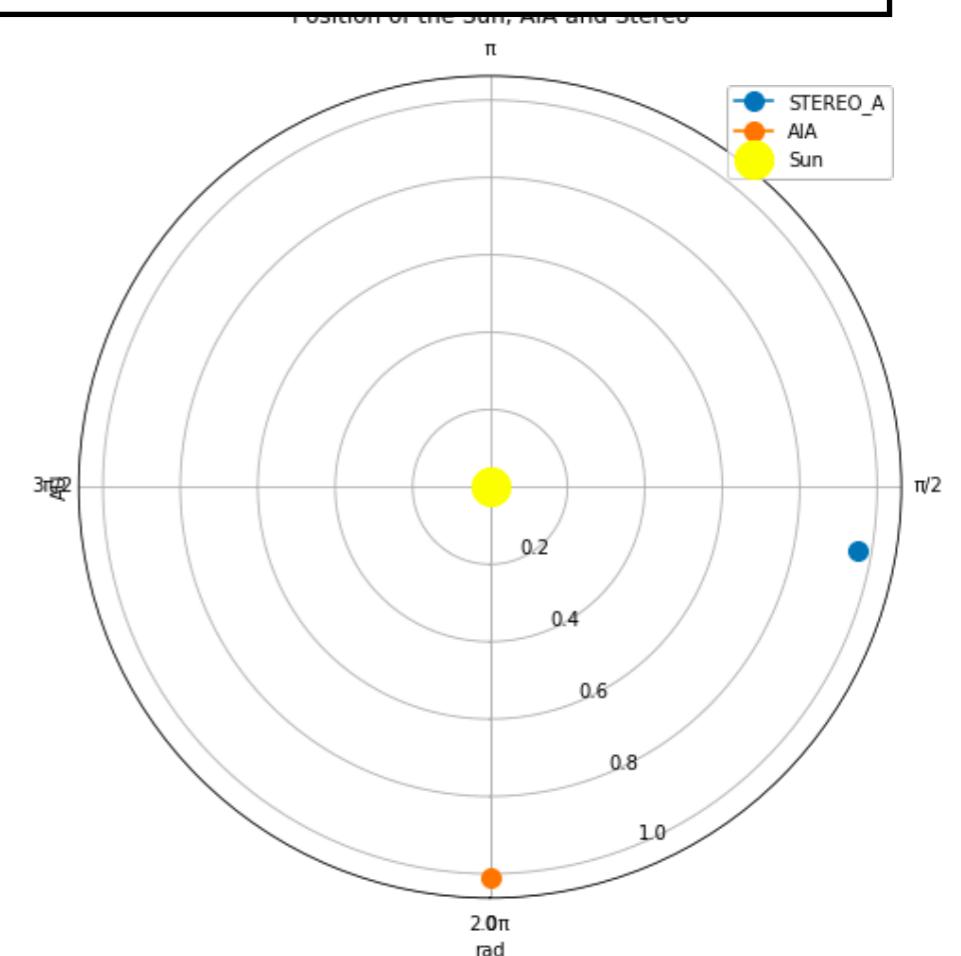


Coordinates

Finding regions of interest



Really important to have this functionality for new observations with Solar Orbiter and Parker Solar Probe!



Positions of satellites



Documentation

User Docs

<https://docs.sunpy.org/en/stable/>



SunPy About ▾ Documentation ▾ Blog Support Us Get Help SunPy Project ▾

SunPy 1.1.2.post1

Search

User's Guide

- Installation
- Brief Tour
- Data Acquisition
- Data Types
- Plotting
- Units and Coordinates
- Time
- Regions of Interest
- Customizing SunPy
- Logging system
- SSWIDL/SunPy Cheat Sheet
- Troubleshooting and Bugs

User's Guide

Welcome to the user guide for SunPy. SunPy is a community-developed, free and open-source solar data analysis environment. It is meant to provide the core functionality and tools to analyze solar data with Python. This guide provides a walkthrough of the major features in SunPy. For more details checkout the [Code Reference](#).

- Installation
 - [Installing Scientific Python and SunPy](#)
 - [Installing SunPy on top of Anaconda](#)
 - [Updating SunPy to a New Version](#)
 - [Advanced SunPy Installation](#)
 - [Advanced Installation Instructions](#)
 - [Testing SunPy](#)
 - [SunPy's Requirements](#)
- Brief Tour
 - [Sample Data](#)
 - [Maps](#)
 - [TimeSeries](#)
 - [Plotting](#)
 - [Solar Physical Constants](#)
 - [Quantities and Units](#)
 - [Working with Times](#)
 - [Obtaining Data](#)
 - [Database Package](#)

v: stable ▾



Documentation

Example Gallery

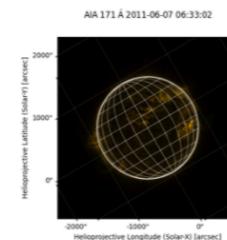
<https://docs.sunpy.org/en/stable/>

The screenshot shows a web browser window with the URL <https://docs.sunpy.org/en/stable/generated/gallery/index.html>. The page title is "Map". The main content area displays several solar maps generated by SunPy, each with a caption below it. The left sidebar contains a navigation menu with links to "User's Guide", "Code Reference", "Example Gallery" (which is highlighted in orange), "Using Remote Data Manager", "Acquiring Data", "Map", and "Plotting". The top navigation bar includes links for "SunPy", "About", "Documentation", "Blog", "Support Us", "Get Help", and "SunPy Project".

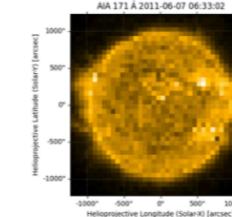
The screenshot shows the sidebar of the SunPy 1.1.2.post1 documentation. The sidebar title is "SunPy 1.1.2.post1". It features a search bar and a navigation menu with the following items: "User's Guide", "Code Reference", "Example Gallery" (highlighted in orange), "Using Remote Data Manager", "Acquiring Data", "Map", "Plotting", "Resampling Maps", and "Finding the brightest pixel".

Map

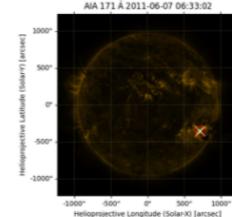
This section contains any examples which showcase how SunPy's [Map](#) can be used to with solar data.



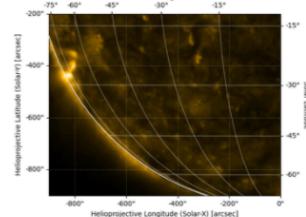
[Rotating a Map](#)



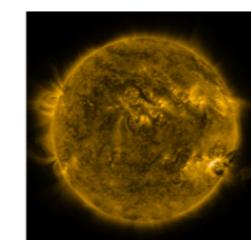
[Resampling Maps](#)



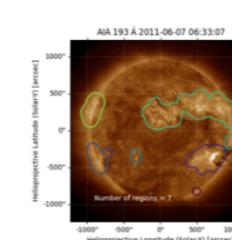
[Finding the brightest pixel](#)



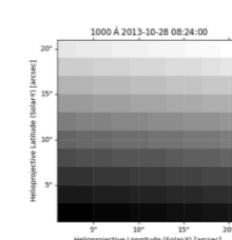
[Cropping a Map](#)



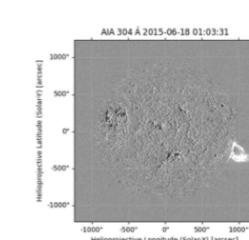
[Plotting a Map without any Axes](#)



[Finding bright regions with ndimage](#)



[Generating a map from data array](#)



[Plotting a difference image](#)

v: stable ▾

Documentation

Example Gallery



<https://docs.sunpy.org/en/stable/>

Go and try it
out!

SunPy 1.1.2.post1

Search

User's Guide

Code Reference

Example Gallery

Using Remote Data Manager

Acquiring Data

Searching and downloading from the VSO

Downloading and plotting LASCO C3 data

Downloading and plotting an HMI magnetogram

Sample data set overview

Map

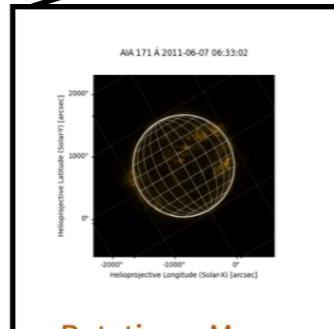
Rotating a Map

Resampling Maps

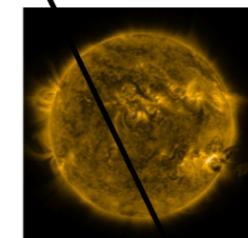
Finding the brightest pixel

Map

This section contains any exa



Rotating a Map



Plotting a Map without
any Axes

Click [here](#) to download the full example code

Rotating a Map

How to rotate a map.

```
import astropy.units as u
import matplotlib.pyplot as plt
import sunpy.map
import sunpy.data.sample
```

We start with the sample data

```
aia_map = sunpy.map.Map(sunpy.data.sample.AIA_171_IMAGE)
```

GenericMap provides the **rotate** method which accepts an angle. This returns a rotated map and does not rotate in place. The data array size is expanded so that none of the original data is lost due to clipping. Note that subsequent rotations are not compounded. The map is only rotated by the specified amount from the original maps orientation.

```
aia_rotated = aia_map.rotate(angle=30 * u.deg)
```



Affiliated Packages

SunPy affiliated packages

Package Name	Description	Documentation	Maintainer
ndcube 	A base package for multi-dimensional (non)contiguous coordinate-aware arrays	ndcube docs	Daniel Ryan
drms	Access HMI, AIA and MDI data with Python	drms docs	Kolja Glogowski
radiospectra	This package aims to provide support for some type of radiospectra on solar physics	radiospectra docs	David Pérez-Suárez
IRISPy sunraster *	A package for handling data from the IRIS satellite	IRISPy docs	Daniel Ryan

Generalized to
slit-spectrograph instruments



Affiliated Packages

SunPy affiliated packages

Package Name	Description		
ndcube		A base package for multi-dimensional (non)contiguous coordinate-aware arrays	ndcube docs
drms		Access HMI, AIA and MDI data with Python	drms docs
radiospectra		This package aims to provide support for some type of radiospectra on solar physics	radiospectra docs
IRISPy sunraster *		A package for handling data from the IRIS satellite	IRISPy docs

Check up hack-day discussion lead by Monica Bobra about review process for affiliated packages!

Generalized to slit-spectrograph instruments

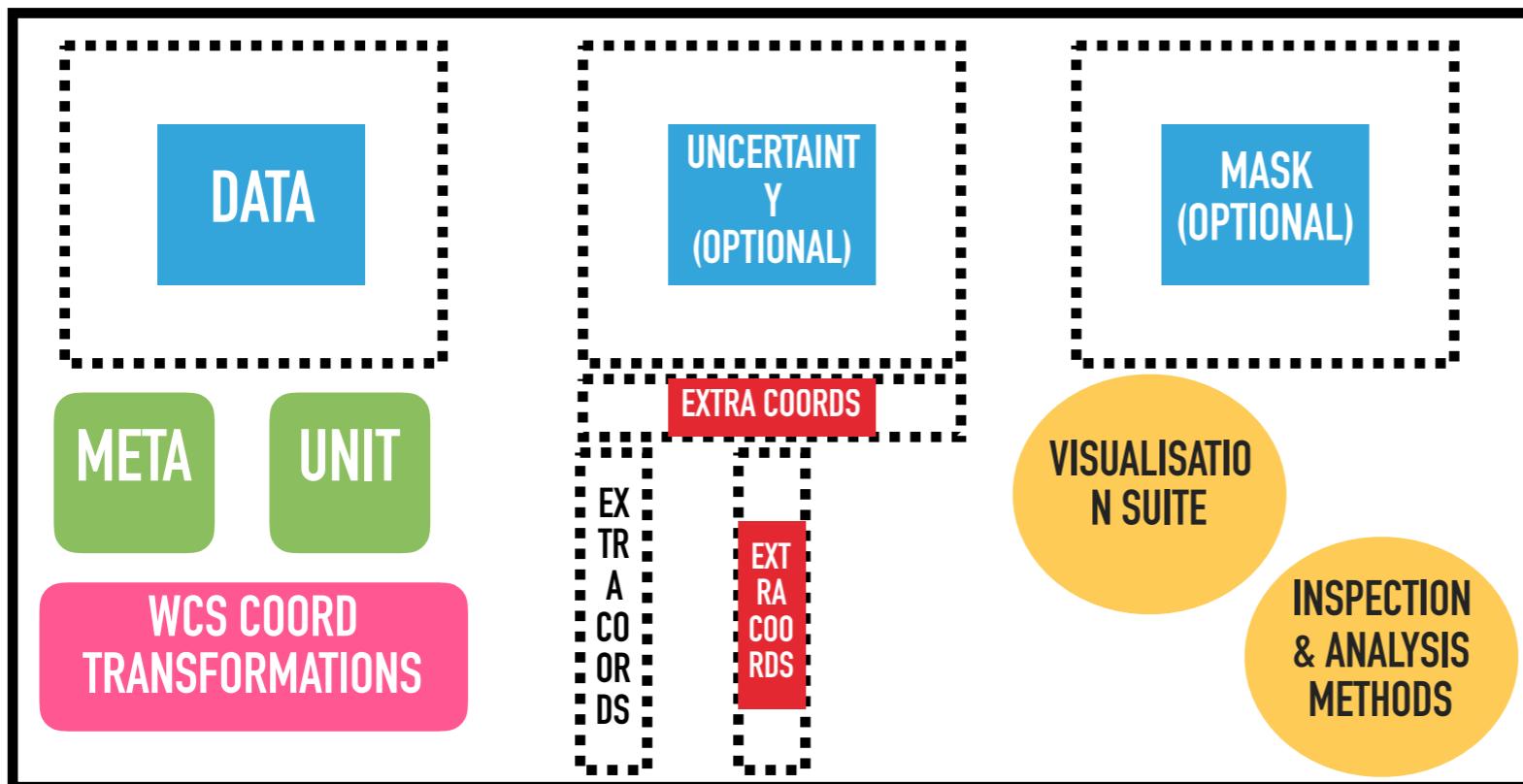


NDCube

SunPy affiliated package

- Provides base classes for ND astronomical and solar data
- Unified slicing, coordinate transformation, data mask, unit, uncertainties, visualization
- Planned for base class for Map in SunPy 3.0

NDCube



Lead developer
Dan Ryan

<https://github.com/sunpy/ndcube>



Looking ahead

Roadmap and future plans

- Two new releases now planned per year - SunPy 2.0 in May ☀️
- Future development and roadmap plan:



- NDCube for Map
- Improved support for data with spectral axes and multi-dimensional data sets
- standardized approach to metadata
- Package template for affiliated packages - incubator for instrument teams?
- Hope to **grow community involvement** - feedback from users, users —> contributors





Get Involved

Please!

- Get involved in the community and contribute!
- Don't need to contribute code - give feedback, report bugs, request features and examples!



sunpy.org

<https://github.com/sunpy/sunpy>

The screenshot shows the SunPy website's navigation bar at the top with links for SunPy, About, Documentation, Blog, Support Us, Get Help, and SunPy Project. Below the navigation bar is a sidebar on the left with links for User's Guide, Code Reference, Example Gallery, Developer's Guide, and Newcomers' Guide (which is highlighted). The main content area is titled "Newcomers' Guide" and contains text about the SunPy newcomers' guide, encouraging contributions, and information about the community and documentation.

SunPy 2.0.dev730+g2af754803

User's Guide
Code Reference
Example Gallery
Developer's Guide
Newcomers' Guide
How to Contribute to SunPy
Not Code
Code
Coding Standards

Newcomers' Guide

Welcome to the SunPy newcomers' guide. If you have come across this page, you just might be new to SunPy. We aim to be a comprehensive Python package that allows solar physicists to deep dive in the vast amount of solar data available.

Firstly, we want to thank you for your interest in contributing to SunPy! SunPy is an open project that encourages everyone to contribute in any way possible.

The people who help develop or contribute to SunPy are varied in ability and experience with the vast majority being volunteers who dedicate time each week. We pride ourselves on being a welcoming community and we would love to have you become a part of our community.

Although this document mainly focuses on how to make contributions to SunPy's code and documentation, there are other ways to get involved with the SunPy community.

If you have any questions, comments or just want to say hello, we have online chat on [Matrix](#) which requires no registration or a [Google Group](#) which you message.

- Get in touch on Github, mailing list, matrix channel or this week!