The SunPy Ecosystem

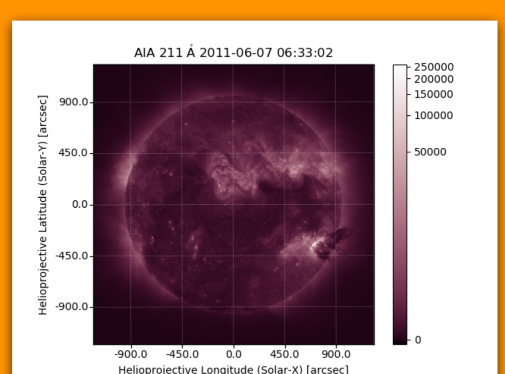
Jack Ireland[1], Stuart Mumford[2], Nabil Freij[3], Daniel Ryan[4], Steven Christe[1], Albert Y. Shih[1], W. T. Barnes[5], A. I. Hamilton[6], SunPy Community

1. NASA Goddard Space Flight Center, Greenbelt, MD, USA. 2. University of Sheffield, Sheffield, UK. 3. Universitat de les Illes Balears, Palma, Spain. 4. USRA, USA. 5. Rice University, TX, USA. 6. University of Hull, UK.

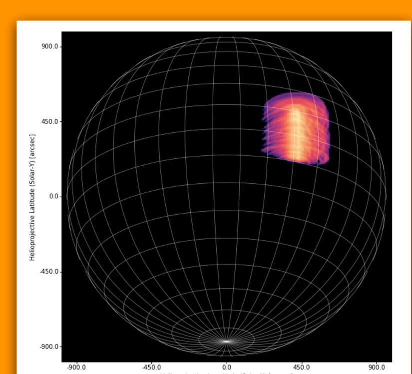
SunPy

SunPy (SunPy Community et al., 2015) is a Python-based project that **enables the analysis of solar data** in the scientific Python data analysis environment. It is built using capabilities from other packages such as Astropy, Numpy, Scipy, pandas and matplotlib. Some of SunPy's features are highlighted below. See **sunpy.org** for more.

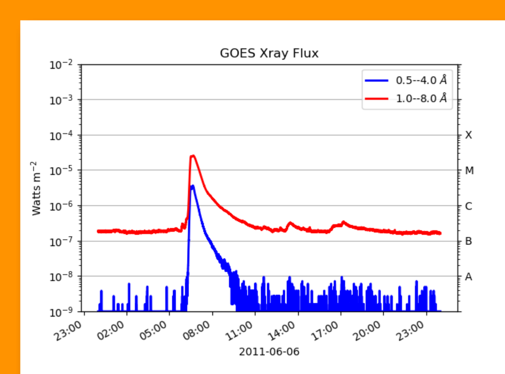
Data objects `Map` and `TimeSeries` make it easy to hold and manipulate the same types of data from different instruments.



co-ordinate aware spatial data



re-projected coronal loop simulations using `Map`



uses `pandas` DataFrames

SunPy's `Fido` **data search & download client** now uses the `drms` package to query the petabyte-scale Solar Dynamics Observatory Joint Science Operations Center (SDO JSOC) image archive.

```
from sunpy.net import Fido, attrs as a
result = Fido.search(a.jsoc.Time('2014-01-01T00:00:00', '2014-01-01T01:00:00'),
                     a.jsoc.Series('hmi.v_45s'), a.jsoc.Notify('my@myemailaddress.org'))
print(result)

Results from 1 Provider:

81 Results from the JSOCClient:
         T_REC          TELESCOP  INSTRUME  WAVELNTH CAR_ROT
         str23            str7     str10    float64   int64
----------------------- -------- ---------- -------- -------
2014.01.01_00:00:45_TAI  SDO/HMI HMI_FRONT2   6173.0    2145
2014.01.01_00:01:30_TAI  SDO/HMI HMI_FRONT2   6173.0    2145
2014.01.01_00:02:15_TAI  SDO/HMI HMI_FRONT2   6173.0    2145
2014.01.01_00:03:00_TAI  SDO/HMI HMI_FRONT2   6173.0    2145
```

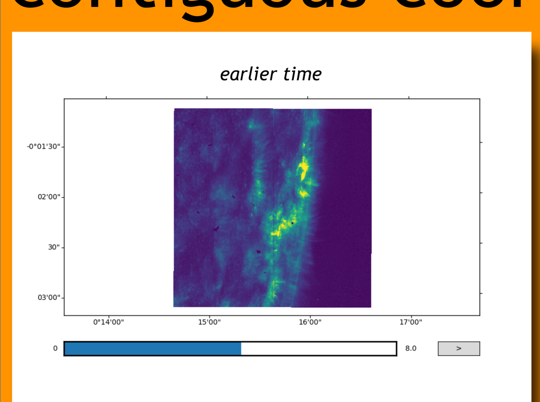
Transformation between solar and astrophysical co-ordinate systems is simple, implemented using `Astropy coordinates`.

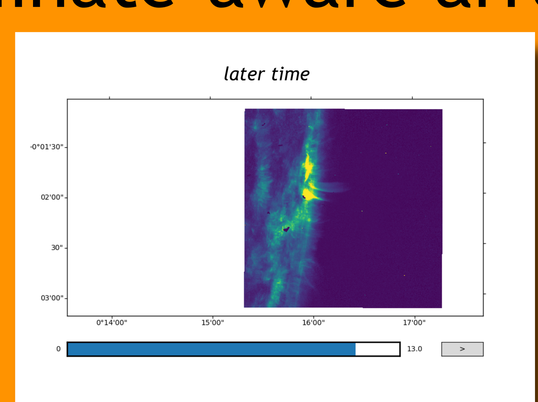
```
import astropy.units as u
from astropy.coordinates import SkyCoord
import sunpy.coordinates
import sunpy.map
from sunpy.data.sample import AIA_193_IMAGE
m = sunpy.map.Map(AIA_193_IMAGE)
point = SkyCoord(123*u.arcsec, 456*u.arcsec, frame=m.coordinate_frame)
print(point) # Helioprojective Cartesian
print(point.transform_to('heliographic_stonyhurst')) # Heliographic Stonyhurst
print(point.transform_to('icrs')) # International Celestial Reference System

<SkyCoord (Helioprojective: obstime=2011-06-07 06:33:07.840000, rsun=696000000.0 m, observer=<HeliographicStonyhurst
Coordinate (obstime=2011-06-07 06:33:07.840000): (lon, lat, radius) in (deg, deg, m)
    (0., 0.048602, 1.51846027e+11)>): (Tx, Ty) in arcsec
    (123., 456.)>
<SkyCoord (HeliographicStonyhurst: obstime=2011-06-07 06:33:07.840000): (lon, lat, radius) in (deg, deg, km)
    (8.50035688, 28.75970887, 695999.9999963)>
<SkyCoord (ICRS): (ra, dec, distance) in (deg, deg, km)
    (230.18600357, 2.18397588, 947137.58662574)>
```

SunPy Ecosystem

Use of SunPy has prompted the development of packages (some of which are outlined below) that support other types of solar physics studies. Efforts are also underway **now** to agree a framework leading to **interoperable data analysis packages for all heliophysical domains** (heliopython.org).

`IRISpy` provides support for data from the Interface Region Imaging Spectrograph (IRIS) via the `ndcube` package. `ndcube` supports multi-dimensional contiguous and non-contiguous coordinate-aware arrays.

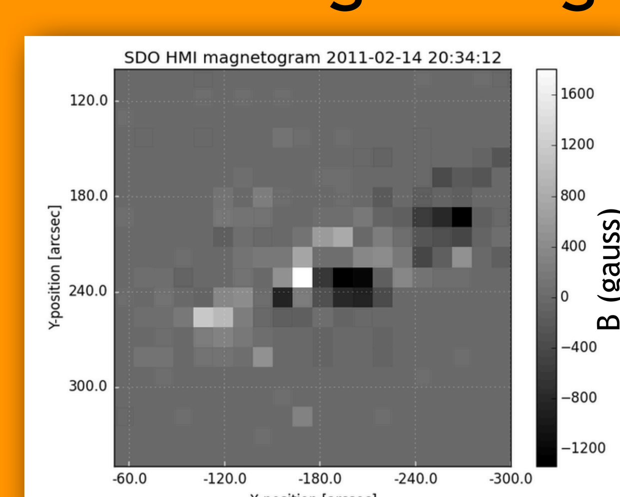




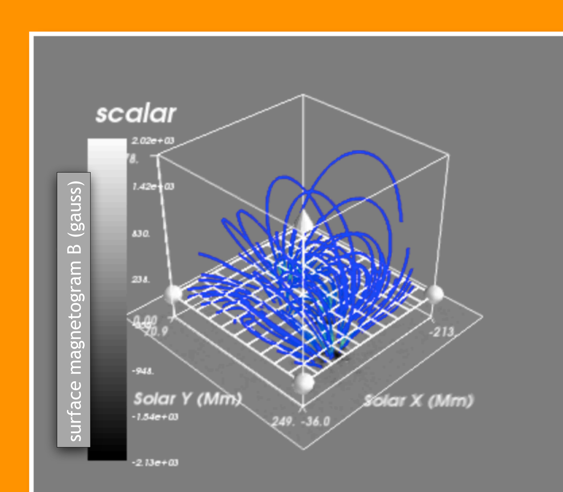
IRIS data is natively 4 dimensional (2 space, time and energy) requiring multi-dimensional coordinate aware arrays for efficient, flexible and intuitive representation and manipulation.

Example 1330Å bandpass IRIS slit jaw images showing rastering over the solar limb, using `IRISpy`. Original data is stored in a three dimensional FITS file.

`solarbextrapolate` provides support for extrapolating the magnetic field of portions of the solar surface from observations of the line-of-sight magnetic flux



downsampled SDO Helioseismic Imager data of the line-of-sight magnetic flux used as input to `solarbextrapolate`

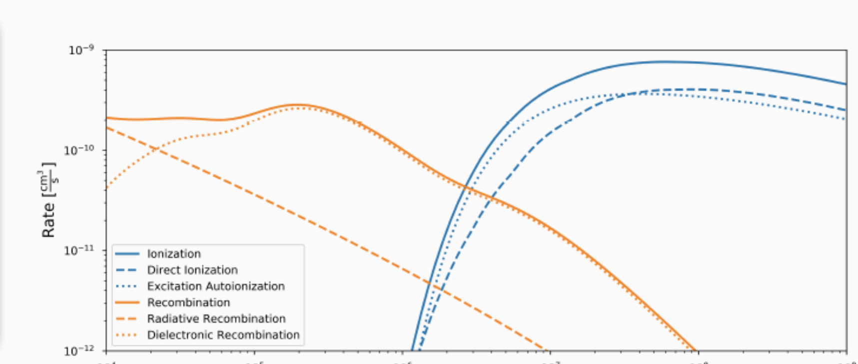


extrapolated magnetic field using the potential field source surface approach

`fiasco` provides support for CHIANTI atomic database - details of emission lines from multiple elements at many ionization stages.

```
>>> import fiasco
>>> import numpy as np; import astropy.units as u
>>> t = np.logspace(4,8,100)*u.K
>>> ion = fiasco.Ion('Fe 16', temperature=t)
# Equivalently
>>> fiasco.Ion('Fe +15', t), fiasco.Ion('iron 16', t);
# Basic metadata
>>> ion.element_name, ion.atomic_symbol, ion.atomic_number
('iron', 'Fe', 26)
>>> ion.ion_name, ion.ionization_stage, ion.charge_state
('Fe 16', 16, 15)
```

```
>>> ion.ionization_rate()
<Quantity [  2.18538687e-258,   1.84291825e-236, ...] cm3 / s>
>>> ion.recombination_rate()
<Quantity [  2.11925041e-10,   2.08729127e-10, ...] cm3 / s>
```



SunPy Community et al., 2015, Computational Science & Discovery, 8(1), 014009

contact: jack.ireland@nasa.gov