sbpy tutorial

Michael S. P. Kelley
LSST SSSC Sprint / 2022 June 6 UTC

Getting started

Getting started 0

sbpy works with Linux and MacOS.

This tutorial includes instructions to setup a clean Python environment with pip.

I am not familiar with conda, but sbpy is available for it. You are free to use a conda environment at your discretion: https://anaconda.org/conda-forge/sbpy

(A conda environment has the benefit of easy installation of pyoorb.)

Alternatively, one can skip Getting Started section and use a RubinSim v0.9 kernel on the the SSSC JupyterHub.

Getting started 1

Create a new (clean) Python environment, activate the environment, and upgrade a few key packages:

```
$ cd sbpy/tutorial/june2022
$ python3 -m venv .venv --prompt=sbpy-tutorial
$ source .venv/bin/activate
$ pip install -U pip setuptools wheel
```

Install JupyterLab with matplotlib widgets:

```
$ pip install jupyterlab ipympl
```

Install sbpy with optional dependencies for a feature-rich experience:

```
$ pip install sbpy[all]
```

https://sbpy.readthedocs.io/en/stable/install.html

Getting started 2 (optional)

Install openorb with Python extensions (pyoorb).

openorb is not pip installable, but for this tutorial we can use one built for sbpy's testing environment:

```
$ pip install git+https://github.com/mkelley/pyoorb-experiment.git@sbpy-testing#egg=pyoorb
```

- It requires gfortran.
- This package is carefully designed for sbpy testing, so it may not work for you.
- It also has a limited DE430 planetary ephemeris file only covering 1990 to 2030.

Getting started 3

What did we just do? We installed and activated an isolated Python environment in a directory named ".venv".

Why did we do that? To ensure that any packages we install do not interfere with your other programs, and to ensure that we are all using similar versions of all necessary packages for this tutorial.

What were some of those packages? `pip freeze` will show them all, e.g., sbpy v0.3.0, astropy v5.1, astroquery v0.4.6, numpy 1.22, scipy 1.8, matplotlib 3.5, etc.

Anything else I should be aware of? This environment takes up ~0.5 GB of space, so you may want to remove the ".venv" directory in the future.

Getting started 4

Start up JupyterLab:

```
$ jupyter-lab
```

Switch to your web browser and start a "Python 3 (ipykernel)". Verify that sbpy is there and ready to go:

```
Python 3.8.11 (default, Aug  1 2021, 12:47:47)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.4.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import sbpy

In [2]: sbpy.__version__

Out[2]: '0.3.0'

In [3]: sbpy.__path__

Out[3]: ['/disks/data0/astro/data/sbpy/tutorial/june2022/.venv/lib/python3.8/site-packages/sbpy']
```

Getting started: an alternative

Start a new notebook with the RubinSim v0.9 kernel on the SSSC JupyterHub.

sbpy basics

All tutorial code is available at:

https://github.com/NASA-Planetary-Science/sbpy-tutorial

and on the SSSC JupyterHub at:

~shared/sbpy-tutorial/sprint-2022

The code blocks in this presentation are snippets from LSST SSSC 2022 tutorial notebooks.

It is recommended to follow the presentation and/or these slides, and refer to the notebooks as needed. Slides also have links to online documentation for reference.

Data containers

sbpy data containers are designed around common needs for solar system astronomy. They encapsulate:

- Ephemerides (sky coordinates, heliocentric distance, etc.)
- Orbits (semi-major axis, time of perihelion, etc.)
- Physical properties (radius, albedo, etc.)
- Obs (astrometry, photometry)

Appropriate units are enforced on all physical quantities (e.g., units of length for radius), and I/O convenience methods are provided for some major online tools and openorb.

The containers are essentially data tables with column name aliases, e.g., heliocentric distance may be accessed via "r" or "rh".

https://sbpy.readthedocs.io/en/stable/sbpy/data/index.html

Ephemerides with Ephem

1. Create an ephemeris object from a dictionary of quantities: rh, delta, and phase: sbpy.data.Ephem.from_dict
2. Use the Minor Planet Center's Ephemeris Service to generate an ephemeris for comet 2P/Encke for the next year: Ephem.from_mpc
3. Display the results in the notebook: Ephem.table.show_in_notebook

```
import astropy.units as u
from astropy.time import Time
from sbpy.data import Ephem

epochs = {'start': Time('2022-06-22'),
          'stop': Time('2023-06-22'),
          'step': 10 * u.day}

eph = Ephem.from_mpc('2P',
                     epochs=epochs,
                     location='I11')
```

https://sbpy.readthedocs.io/en/stable/sbpy/data/ephem.html

Orbits and OpenOrb

1. Use the Minor Planet Center to get the orbit of comet 2P/Encke: sbpy.data.Orbit.from_mpc
2. Convert the orbit into an ephemeris over the next year: Ephem.from_oo

```
import astropy.units as u
from astropy.time import Time
from sbpy.data import Orbit, Ephem


orbit = Orbit.from_mpc('2P')
orbit['targetname'] = '2P'
orbit['orbtype'] = 'COM'
orbit['H'] = 15 * u.mag
orbit['G'] = 0.15
epochs = Time('2026-01-01') + np.arange(365, step=30) * u.day
eph = Ephem.from_oo(orbit, epochs=epochs, location='I11')
```

https://sbpy.readthedocs.io/en/stable/sbpy/data/orbit.html

Activities

1. Get the ephemeris of your favorite comet or asteroid from JPL Horizons or the IMCCE's Miriade: Ephem.from_horizons, Ephem.from_miriade
2. Get the physical properties of an asteroid or three from the JPL Small-Bodies Database: sbpy.data.Phys.from_sbdb

We will use the observational data object, sbpy.data.Obs, later in this tutorial.

```
print(phys['targetname', 'H', 'D', 'albedo'])
<QTable length=3>
        targetname            H    diameter  albedo
                             mag      km
          str26            float64 float64  float64
------------------------- ------- -------- -------
         1 Ceres (A801 AA)    3.56    939.4    0.09
 12893 Mommert (1998 QS55)   13.98    5.214   0.179
3552 Don Quixote (1983 SA)   12.96     19.0    0.03
```

https://sbpy.readthedocs.io/en/stable/sbpy/data/phys.html

Photometric calibration

sbpy photometric calibrations are based on spectra of the Sun and Vega, precomputed filter zeropoints, and relevant bandpass wavelengths.

Magnitudes and physical units

1. Convert an AB magnitude to/from linear units: astropy.units.ABmag, astropy.units.spectral_density
2. Convert a Vega magnitude into a physical unit: sbpy.units.VEGAmag, sbpy.units.spectral_density_vega

```
import astropy.units as u


m = 18 * u.ABmag

fnu = m.to('W/(m2 Hz)')

flam = m.to('W/(m2 um)', u.spectral_density(550 * u.nm))
```

https://docs.astropy.org/en/stable/units/equivalencies.html
https://sbpy.readthedocs.io/en/stable/sbpy/units.html#vega-magnitudes

Bandpasses

1. Plot the PS1 r-band bandpass: sbpy.photometry.bandpass, SpectralElement.plot
2. Convert an AB magnitude measured in the PS1 r-band into a linear unit: ABmag, spectral_density, SpectralElement.pivot

```
import astropy.units as u


m = 18 * u.ABmag

fnu = m.to('W/(m2 Hz)')

flam = m.to('W/(m2 um)', u.spectral_density(550 * u.nm))
```

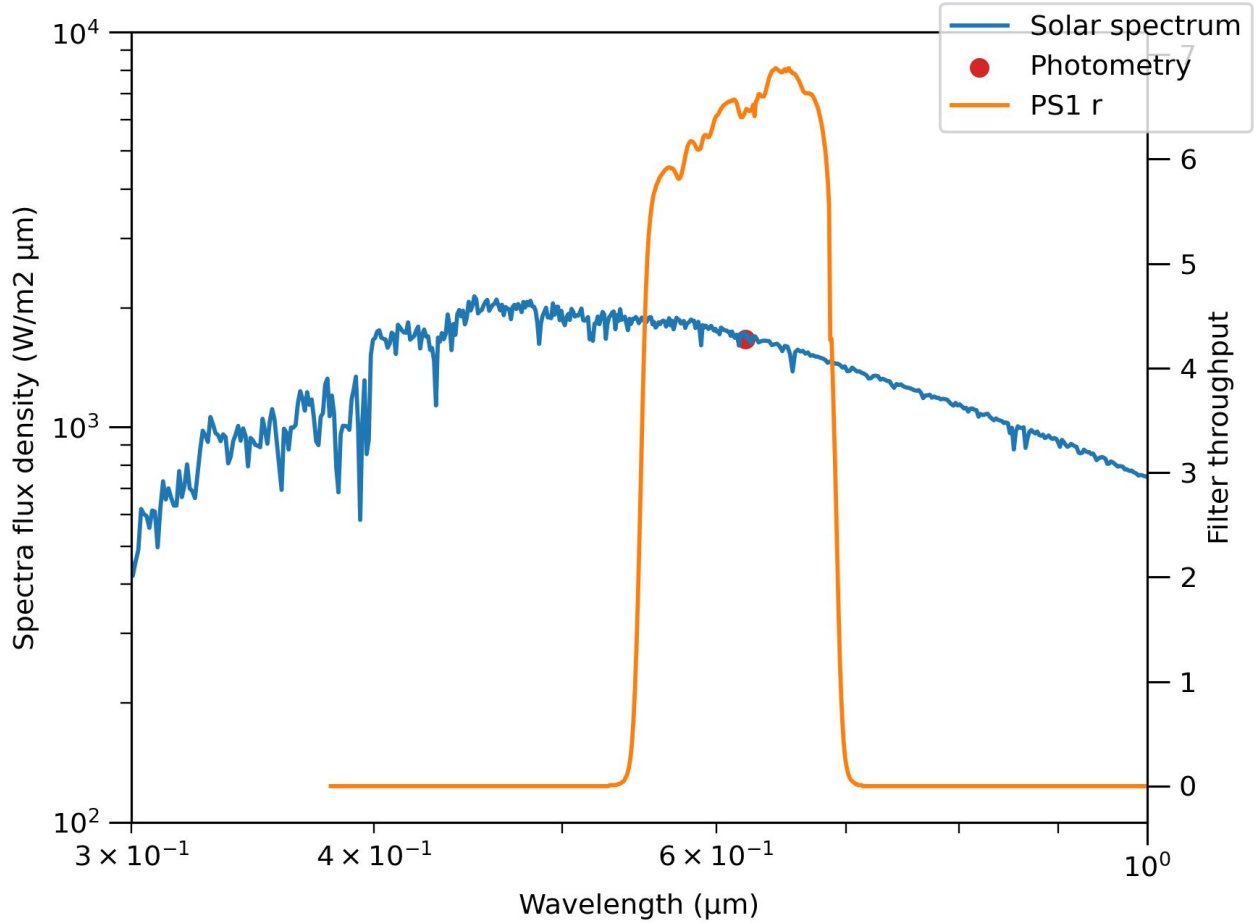
https://sbpy.readthedocs.io/en/stable/sbpy/photometry.html#filter-bandpasses
https://synphot.readthedocs.io/en/latest/synphot/bandpass.html

Solar spectrum

1. Observe the Sun through in a V-band filter: bandpass, sbpy.calib.Sun.from_default, Sun.observe_bandpass
2. Plot the spectrum of the Sun, the brightness of the Sun measured in (1), and the PS1 r-band throughput: Sun.wave, Sun.fluxd, SpectralElement.waveset, SpectralElement()
3. Get the apparent magnitude of the Sun in the LSST r-band filter, as calculated by Willmer (2018): sbpy.calib.solar_fluxd, Sun.observe_filter_name

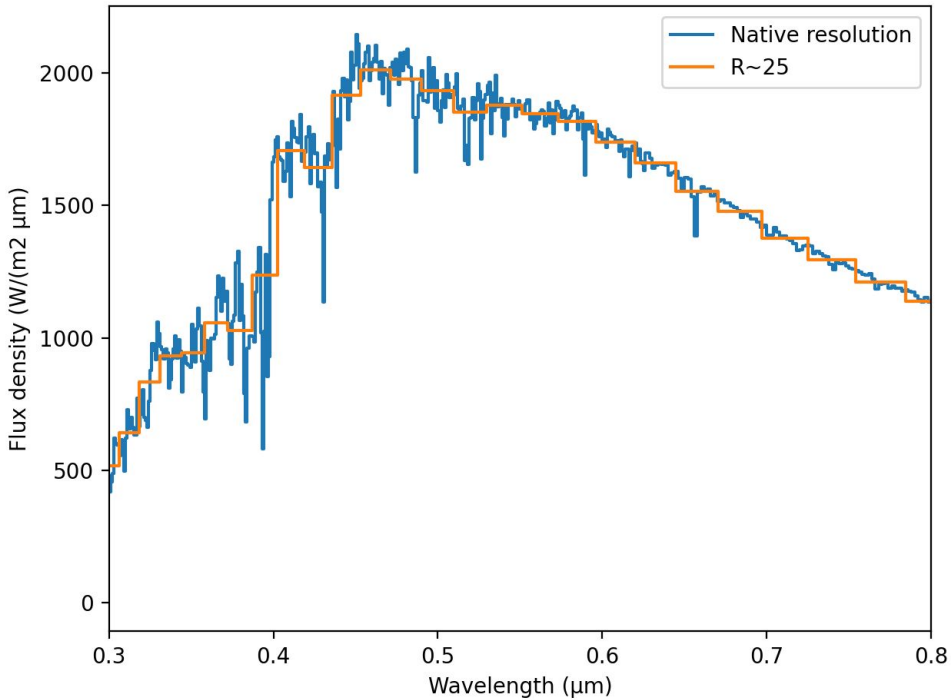
```
from sbpy.calib import Sun

from sbpy.photometry import bandpass


sun = Sun.from_default()

V = bandpass('Johnson V')

lambda_eff, fluxd = sun.observe_bandpass(V)
```

https://sbpy.readthedocs.io/en/stable/sbpy/calib.html#observe-the-sun


Solar spectrum
Photometry
PS1 r
Spectra flux density (W/m2 µm)
Filter throughput
Wavelength (µm)
10^4
10^3
10^2
3×10^{-1}
4×10^{-1}
6×10^{-1}
10^0
0
1
2
3
4
5
6

Activities

1. Plot a spectrum of the Sun, and compare it to the same spectrum re-binned to a spectral resolution of 25: Sun.observe
2. Convert an absolute magnitude of 12.436 ABmag in the LSST r-band filter to a cross-sectional area assuming a reflectance of 4% per steradian: astropy.units.Quantity.to, sbpy.units.reflectance



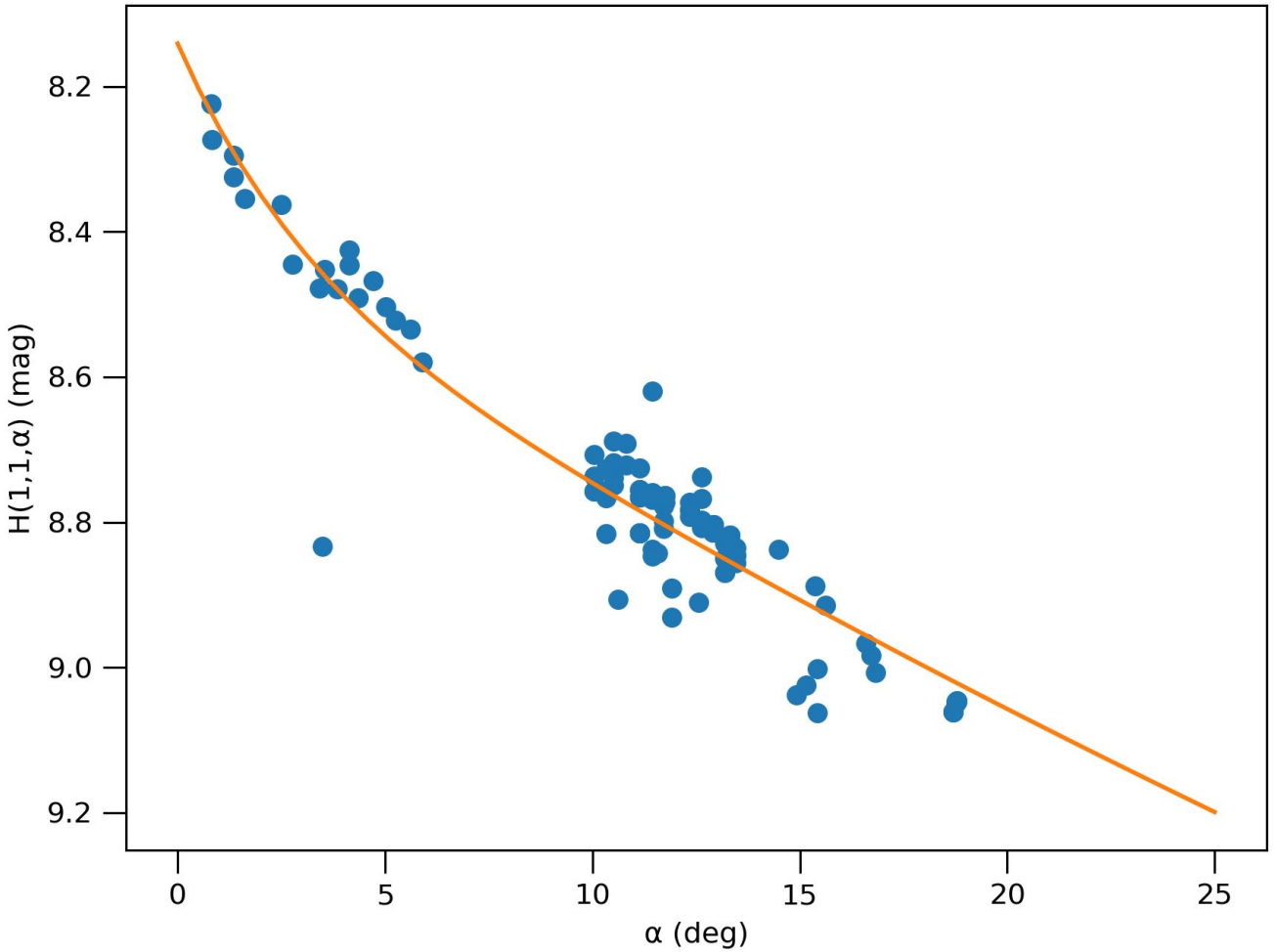
Surfaces!

Disk integrated phase functions

1. Get all reported r-band photometry of (90) Antiope from ZTF, and ephemeris data for each observation: sbpy.data.Obs.from_mpc, Orbit.from_horizons, Ephem.from_oo
2. Convert the photometry to absolute magnitude and fit the data with the H, G photometric model: astropy.modeling.fitting.LevMarLSQFitter, sbpy.photometry.HG

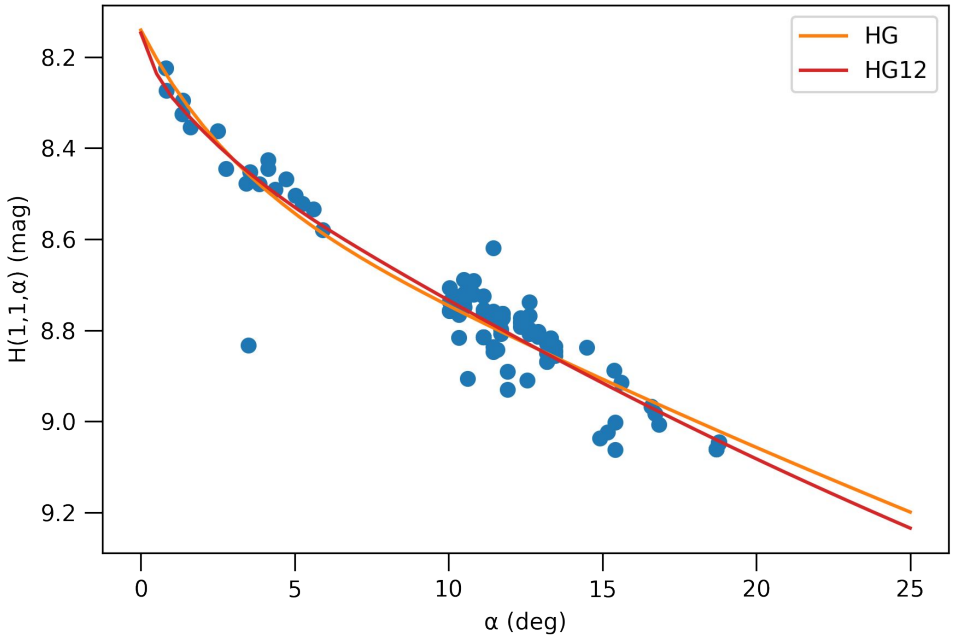
```
from astropy.time import Time
from sbpy.data import Obs, Orbit, Ephem


phot = Obs.from_mpc('90')
ztf = ((phot['observatory'] == 'I41')
       * (phot['epoch'] > Time('2017-10-15')))
r = phot['band'] == 'r'
phot = phot[ztf * r]
orbit = Orbit.from_horizons('90')
eph = Ephem.from_oo(orbit, location='I41',
                    epochs=phot['epoch'])
```

https://sbpy.readthedocs.io/en/stable/sbpy/data/obs.html
https://sbpy.readthedocs.io/en/stable/sbpy/photometry.html#disk-integrated-phase-function-models
https://docs.astropy.org/en/stable/modeling/index.html


8.2
8.4
8.6
8.8
9.0
9.2
H(1,1,α) (mag)
0
5
10
15
20
25
α (deg)

Activities

Fit the (90) Antiope data with the other photometric models and compare the results: HG1G2, HG12, HG1G2_Pen16



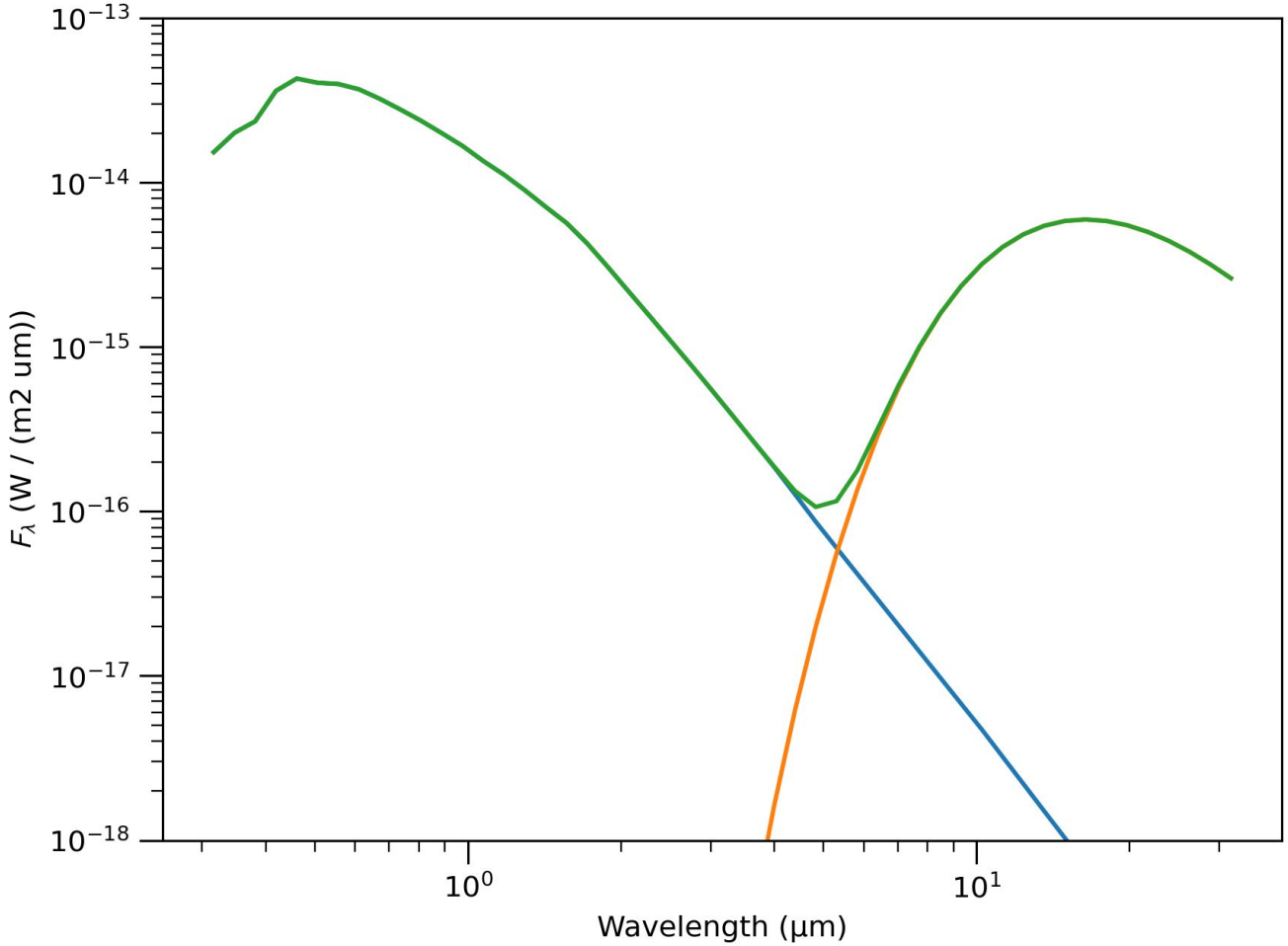
Dust!

Cometary coma dust

1. Create Afρ and εfρ objects: sbpy.activity.dust.Afrho, sbpy.activity.dust.Efrho
2. Convert the Afρ object into flux density, using the LSST r-band filter, a 5" radius aperture, and the observational circumstances of your favorite comet: Afrho.to_fluxd, Ephem.from_mpc
3. Convert that flux density back to the Afρ quantity: Afrho.from_fluxd
4. Plot the 0.3 to 30 μm spectrum of a comet with the Afρ and εfρ objects: Afrho.to_fluxd, Efrho.to_fluxd

```
import astropy.units as u

from sbpy.data import Ephem

from sbpy.activity import Afrho


afrho = Afrho(1000 * u.cm)

eph = Ephem.from_mpc('C/2017 K2')

aper = 5 * u.arcsec

m = afrho.to_fluxd('LSST r', aper, eph, unit=u.ABmag)
```

https://sbpy.readthedocs.io/en/stable/sbpy/activity/dust.html


10^{-13}
10^{-14}
10^{-15}
10^{-16}
10^{-17}
10^{-18}
F_λ (W / (m2 um))
10^{0}
10^{1}
Wavelength (μm)

Activities

1. Account for phase effect in the previous coma spectrum: Afrho.to_fluxd, sbpy.activity.dust.phase_HalleyMarcus
2. Redden the scattered light from the coma by a slope of 10%/100 nm measured at 616.5 nm: sbpy.units.hundred_nm, sbpy.spectroscopy.SpectralGradient, sbpy.spectroscopy.Reddening

```
wave = np.linspace(0.3, 1.0) * u.um

print(reddener(wave))

[0.6835     0.69778571 0.71207143 0.72635714 0.74064286 0.75492857
 0.76921429 0.7835     0.79778571 0.81207143 0.82635714 0.84064286
 0.85492857 0.86921429 0.8835     0.89778571 0.91207143 0.92635714
 0.94064286 0.95492857 0.96921429 0.9835     0.99778571 1.01207143
 1.02635714 1.04064286 1.05492857 1.06921429 1.0835     1.09778571
 1.11207143 1.12635714 1.14064286 1.15492857 1.16921429 1.1835
 1.19778571 1.21207143 1.22635714 1.24064286 1.25492857 1.26921429
 1.2835     1.29778571 1.31207143 1.32635714 1.34064286 1.35492857
 1.36921429 1.3835    ]
```

https://sbpy.readthedocs.io/en/stable/sbpy/activity/dust.html#phase-angles-and-functions
https://sbpy.readthedocs.io/en/stable/sbpy/spectroscopy/index.html

Gas!

Cometary coma gas

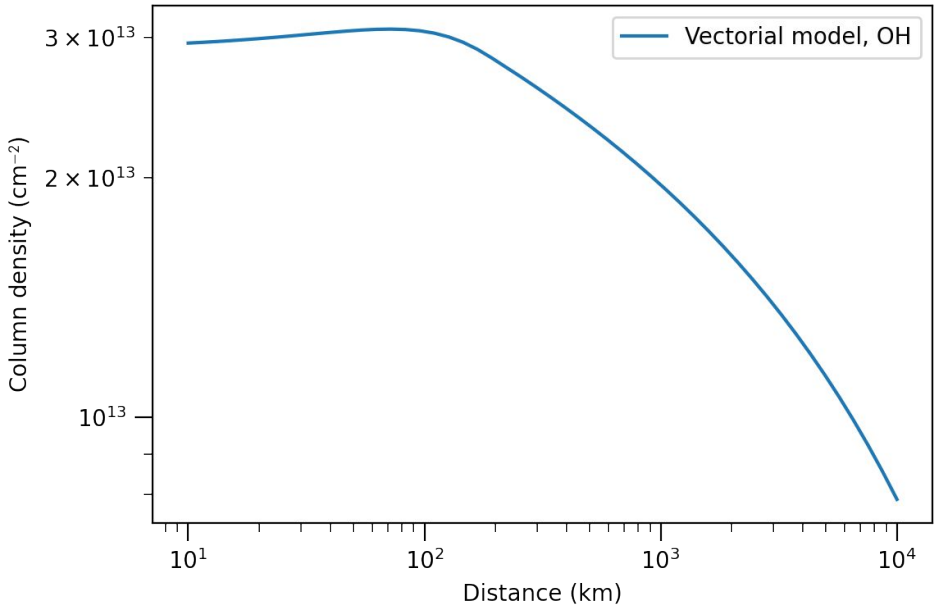
1. Use the Haser (1957) model to compute the total number of water molecules in a coma at 1 au from the Sun, as observed with a 10,000 km radius aperture: sbpy.activity.gas.photo_lengthscale, sbpy.activity.gas.Haser
2. Repeat the calculation, but instead observe with a narrow slit of dimensions 1"×10": sbpy.activity.RectangularAperture, Ephem.from_mpc

```
import astropy.units as u
from sbpy.activity.gas import Haser, photo_lengthscale


Q = 1e28 / u.s
v = 800 * u.m / u.s
parent = photo_lengthscale('H2O')
water = Haser(Q, v, parent)


rho = 1e4 * u.km
print(water.total_number(rho))
```

https://sbpy.readthedocs.io/en/stable/sbpy/activity/gas.html
https://sbpy.readthedocs.io/en/stable/sbpy/activity/index.html#apertures

Activities

1. Get the photolysis lengthscale for OH and use the Haser model to calculate the total number of OH molecules in the 1"×10" slit: photo_lengthscale, Haser
2. Estimate the total luminosity of the OH 0-0 band given the results of (1): fluorescence_band_strength
3. Repeat with the Vectorial model of Festou (1981): sbpy.activity.gas.VectorialModel



https://sbpy.readthedocs.io/en/stable/sbpy/activity/gas.html#fluorescence
https://sbpy.readthedocs.io/en/stable/sbpy/activity/gas.html#vectorial-model

Feedback, Questions, Help

Help

- We have a channel in the astropy slack workspace: Find #sbpy at astropy.slack.com
- Find me in the SSSC or LSST slack workspaces.
- Email the developers, Jian-Yang Li and me: msk@astro.umd.edu and jyli@psi.edu

Problems, bugs, proposed enhancements, etc.

- Open an issue at: https://github.com/NASA-Planetary-Science/sbpy/issues