





Michael Mommert, M. Kelley, M. de Val-Borro, Jian-Yang Li, G. Guzman, B. Sipőcz,
J. Ďurech, M. Granvik, W. Grundy, N. Moskovitz, A. Penttilä, N. Samarasinha

EPSC/DPS 2019 workshop, September 17, 2019

What is this workshop about?

- sbpy!
 - What is it, how can it help you, how can you use it?
 - sbpy is for you: what would you like to see?
- This workshop is not:
 - An introduction to Python programming
 - An attempt to convert you into a Python programmer

Our 30 sec Elevator Pitch

- A Python package for **small-body** planetary astronomy
- A collection of basic functions and methods for asteroid and comet observers/researchers
- Astropy affiliated 
- Current version v0.2, v1.0 release in 2021
- Funded through NASA PDART 

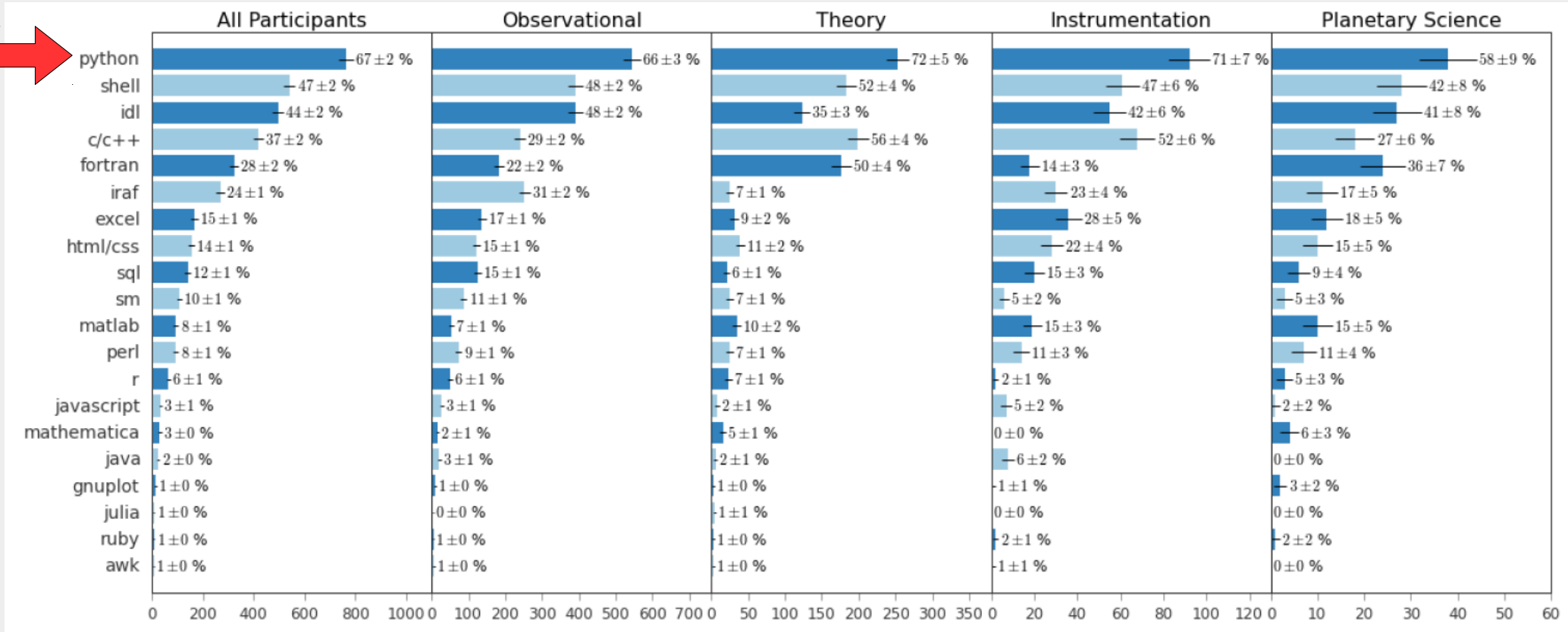
sbpy.org

Motivation

- Imagine... you need some code to solve a problem...
 - Write the code from scratch (time intensive)
 - Use somebody's code (is it reliable?)
 - Re-write somebody's FORTRAN code (ewww...)
 - Use a well-tested and documented existing code
- Astropy (astropy.org)
- Provide especially young researchers with a code base to kick-start their research



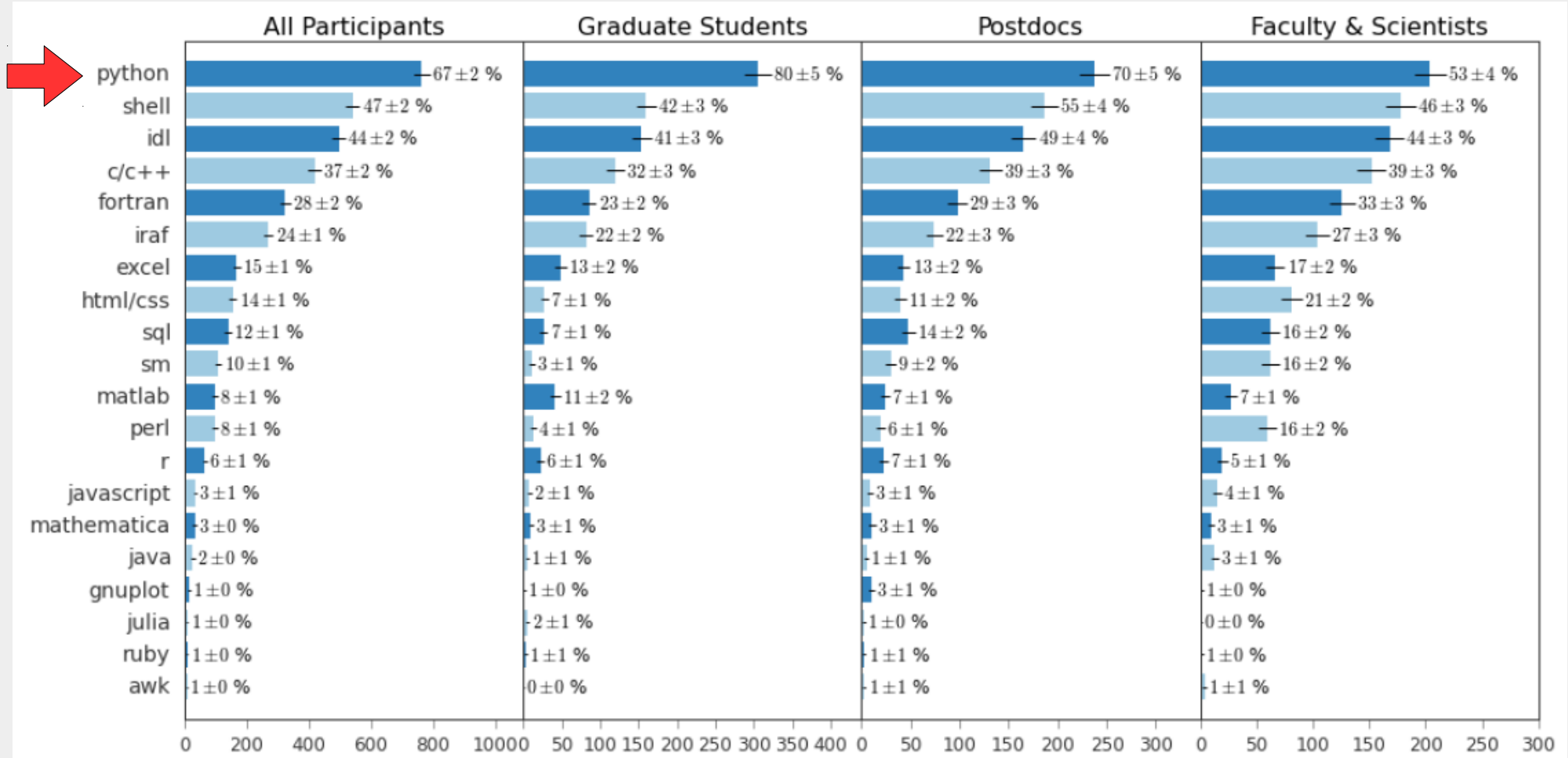
Why Python?



“Which of these programming languages do you frequently use in your research?”

Momcheva and Tollerud (2015)

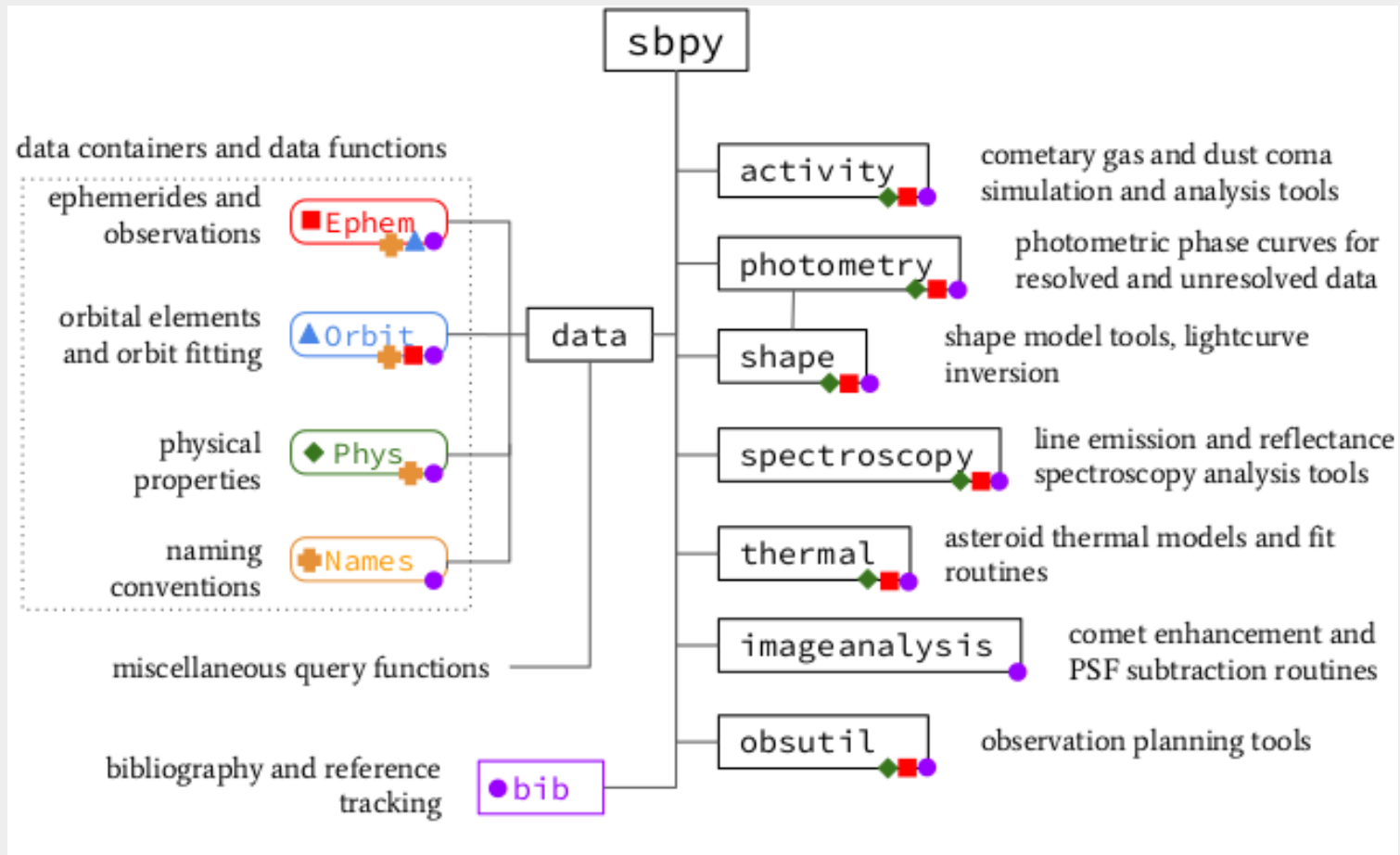
Why Python?



“Which of these programming languages do you frequently use in your research?”

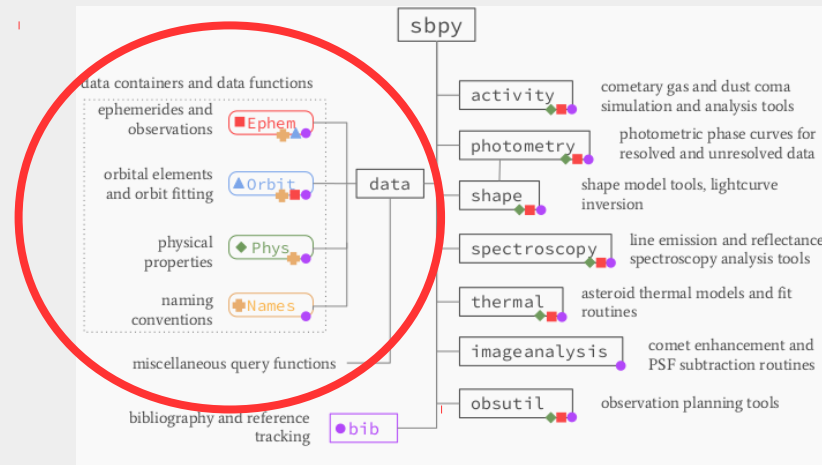
Momcheva and Tollerud (2015)

sbpy - Inventory



sbpy.data

- A collection of data containers:
 - **Orbit**: orbital elements [notebook](#)
 - **Ephem**: ephemerides and observational data [notebook](#)
 - **Obs**: ephemerides and observational data [notebook](#)
 - **Phys**: physical properties [notebook](#)
 - **Names**: asteroid and comet name/number/designation [notebook](#)
- Common functionality (use the same base class) utilizing `astropy.table`
- Data input/output of all sbpy functions uses `sbpy.data`
- Aimed at convenience and flexibility



sbpy.data - Examples

- Different ways to generate data objects:

- from a list or array:

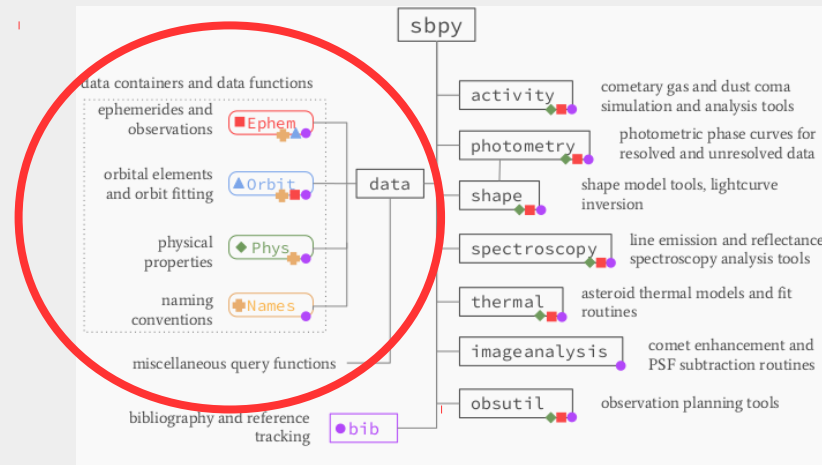
```
>>> eph = Ephem.from_columns(  
...     [[10.2234, 10.233453]*u.deg,  
...     [-12.42123, -12.42562]*u.deg,  
...     [2451234.1234, 2451234.2345]*u.d],  
...     names=['ra', 'dec', 't'])
```

- from a dictionary:

```
>>> ceres = Phys.from_dict({'d':986*u.km, 'pv':0.09})
```

- from an online resource:

```
>>> eph = Ephem.from_horizons('ceres')
```



notebook

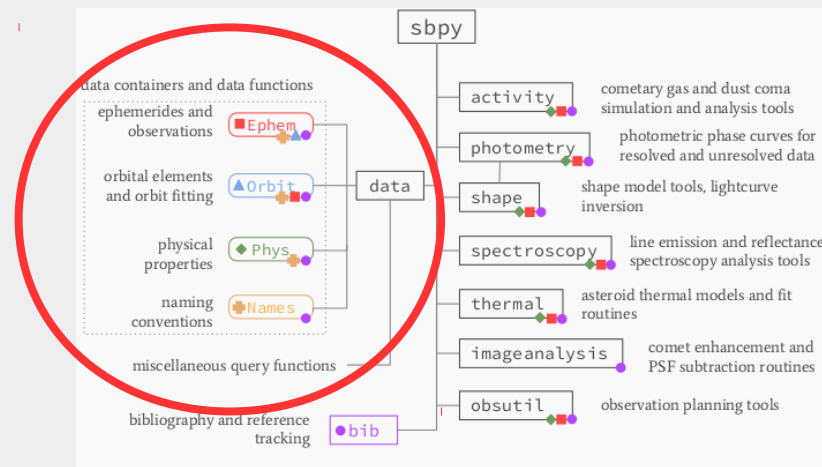
sbpy.data - Examples

- Using sbpy.data objects:

```
>>> eph.field_names  
<TableColumns names=('targetname', 'RA', 'DEC', ... )>
```

```
>>> eph['ra', 'dec']  
<QTable length=10>  
      RA      DEC  
      deg      deg  
-----  
139.60684 26.72378  
140.20138 26.46513  
142.0532  25.66408  
...      ...
```

```
>>> eph['RA_rate'].to('arcsec/s')  
<Quantity [0.01218538, 0.01244878, ...] arcsec / s>
```



sbpy.data

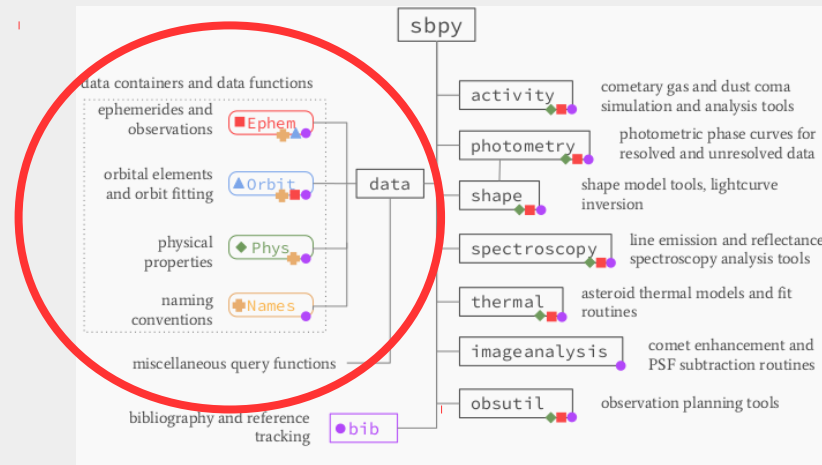
- Home to query functions:

- **JPL Horizons**: ephemerides, orbital elements, and state vectors (using `astroquery.jplhorizons`)

```
Ephem.from_horizons('ceres')
```

[notebook](#)

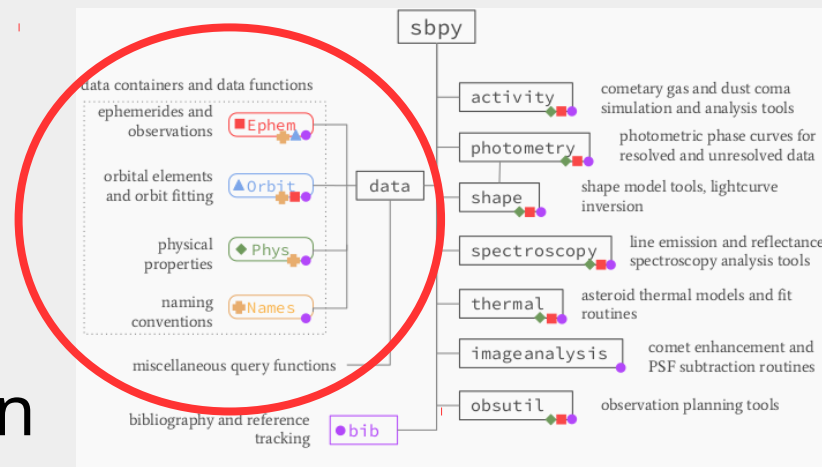
- **JPL SBDB**: physical properties (using `astroquery.jpalsbdb`)
 - **Minor Planet Center** (orbits, ephemerides, observations, using `astroquery.mpc`)
 - **IMCCE** (ephemerides using `astroquery.imcce`)
- Future query functions (mostly as part of `astroquery`):
 - **Lowell Observatory** (ephemerides, physical properties)



sbpy.data

- Interoperability with other Python modules:

- **Pyoorb**: a Python interface to OpenOrb
 - implemented: element transformations, orbit propagation, ephemeris computation
 - Future functions: ranging, orbit fitting
- **SpiceyPy** (tbd): SPICE interface



sbpy.data

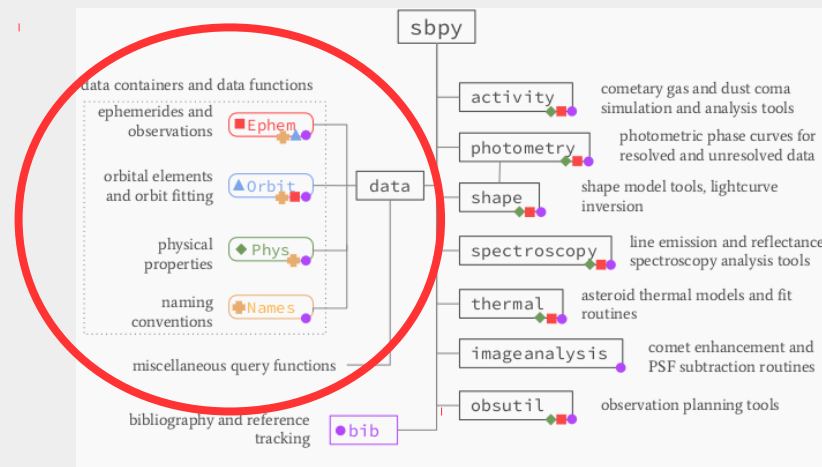
- Name resolving functionality

```
>>> from sbpy.data import Names
```

```
>>> Names.parse_asteroid('3552 Don Quixote (1983 SA)')
{'number': 3552, 'name': 'Don Quixote', 'desig': '1983 SA'}
```

```
>>> Names.parse_comet('72P/Denning-Fujikawa')
{'type': 'P', 'number': 72, 'name': 'Denning-Fujikawa'}
```

```
>>> Names.asteroid_or_comet('72P/Denning-Fujikawa')
'comet'
```



notebook

sbpy.data

- Field name translation

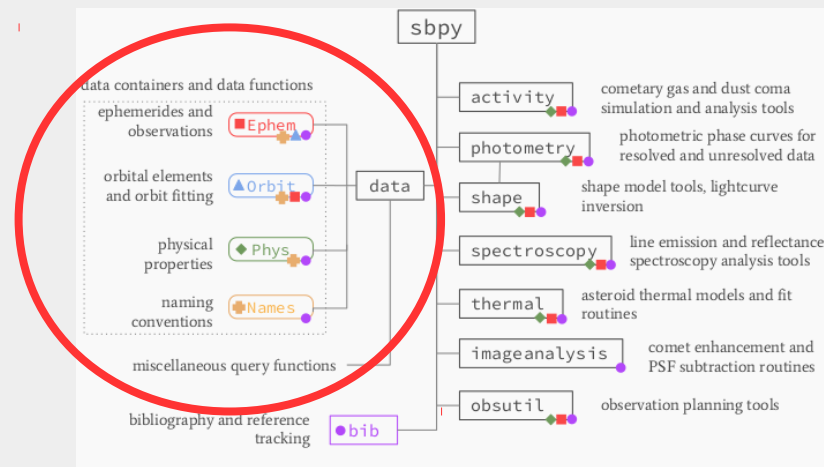
```
>>> from sbpy.data import Phys
>>> import astropy.units as u
```

```
>>> data = Phys.from_dict({'d': 10*u.km})
>>> data['d']
<Quantity [10.] km>
```

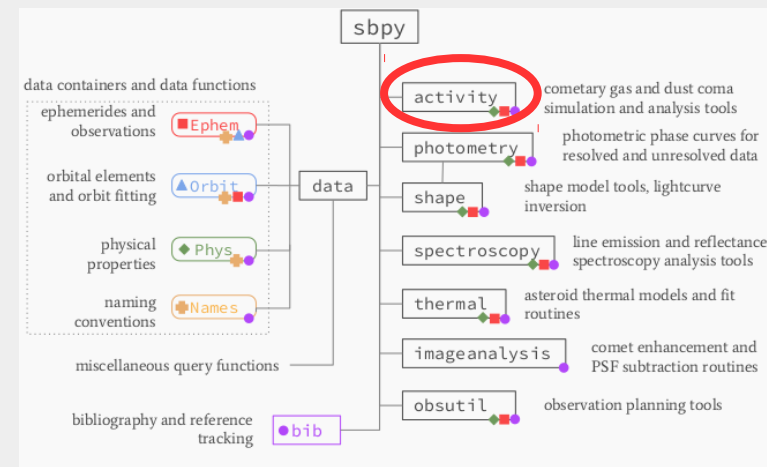
```
>>> data['diameter']
<Quantity [10.] km>
```

- Field conversion

```
>>> data['radius']
<Quantity [5.] km>
```



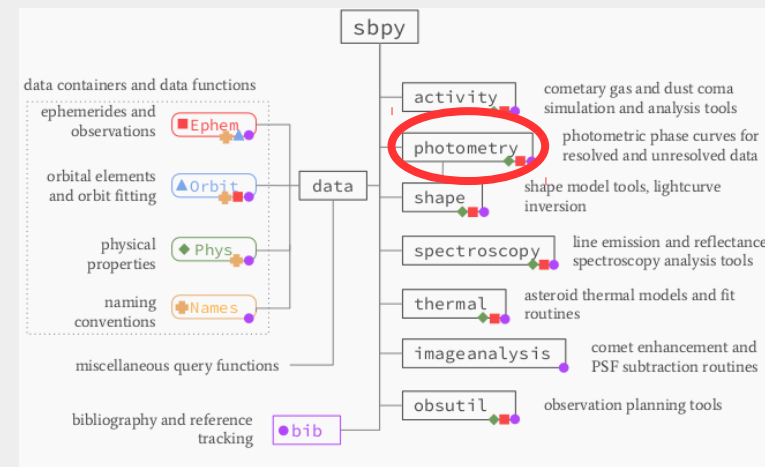
sbpy.activity



- Modeling of cometary comae:
 - Dust activity: Afp (basic functionality implemented) [notebook](#)
 - Gas activity:
 - Haser Model (basic functionality implemented) [notebook](#)
 - Vectorial Model (tbd)
 - Syndyne/Synchrone Model (tbd)
 - Ice sublimation Model (tbd)
 - LTE and non-LTE (using pyradex) production rates [notebook](#)

sbpy.photometry

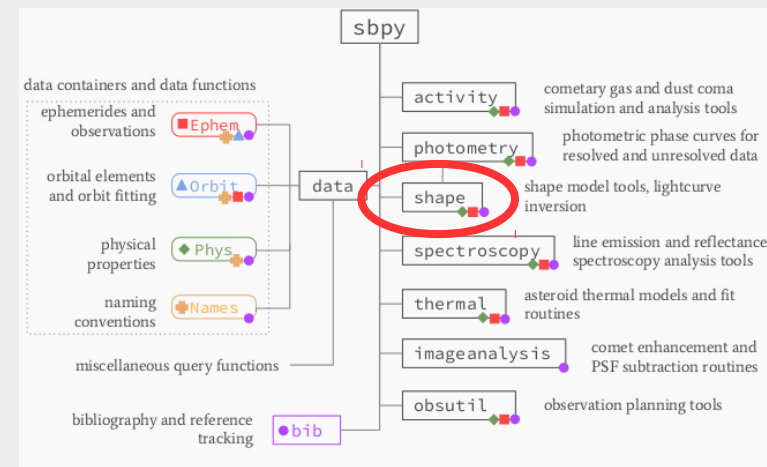
- Implementation of different light-scattering models:
 - HG asteroid phase curve model (basic)
 - HG1G2 asteroid phase curve model (basic)
 - HG12 asteroid phase curve model (basic)
 - Linear phase curve model (basic)
 - Lunar, Lommel-Seeliger, Lunar Lambert, ROLO (basic)
 - Hapke (5-parameter version, spectral mixing, tbd)



notebook

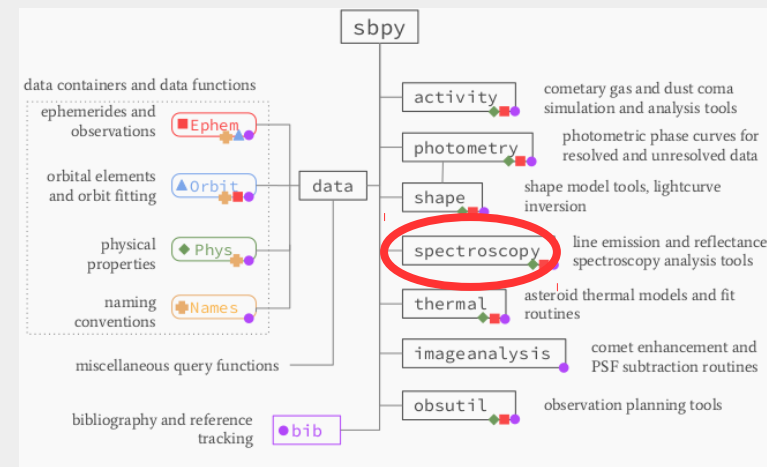
sbpy.shape

- Lightcurve fitting tools (tbd)
- Kaasalainen shape modeling tools ([Ďurech code](#), tbd)
- Lightcurve modeling based on shape models (tbd)



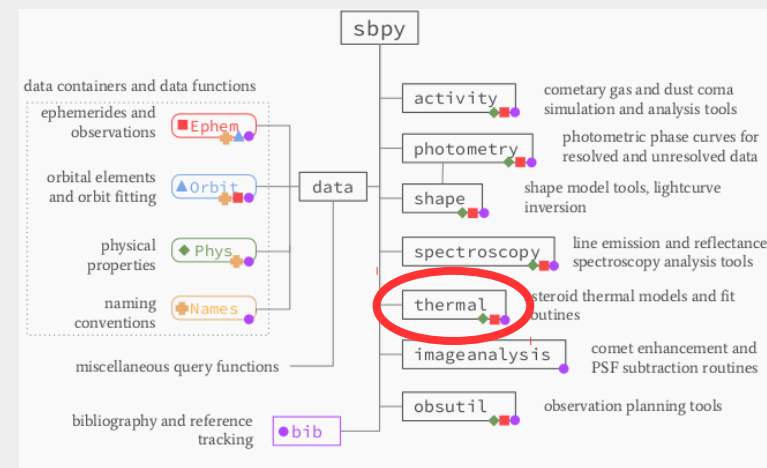
sbpy.spectroscopy

- Tools for querying spectral libraries (e.g., JPL using [astroquery.jplspec](#)) [notebook](#)
- Tools for fitting emission line and reflectance spectra (tbd)
- Spectrophotometry and spectrum convolution (tbd)

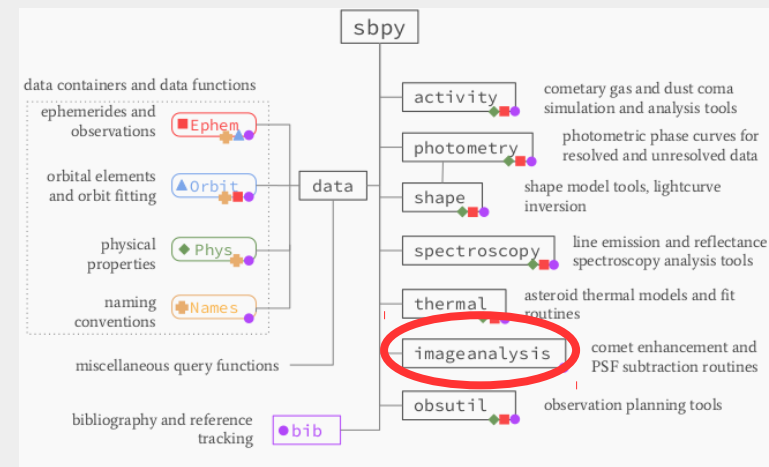


sbpy.thermal

- Thermal modeling capabilities for estimating fluxes and fitting thermal-infrared observations: (tbd)
 - Standard Thermal Model
 - Fast-Rotating Model
 - Near-Earth Asteroid Thermal Model

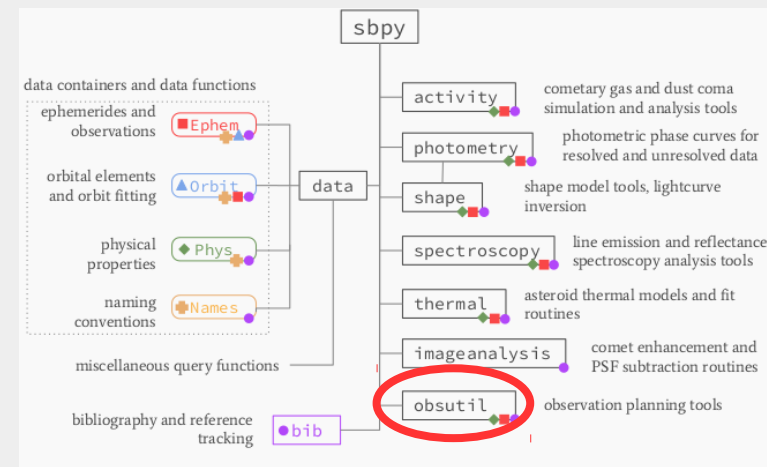


sbpy.imageanalysis



- Cometary coma image enhancement methods (tbd)
- PSF-subtraction tools (tbd)

sbpy.obsutil



- Tools for the planning of asteroid and comet observations (tbd)
 - Identify peak observability
 - Create observing scripts
 - Create finder charts

sbpy.bib

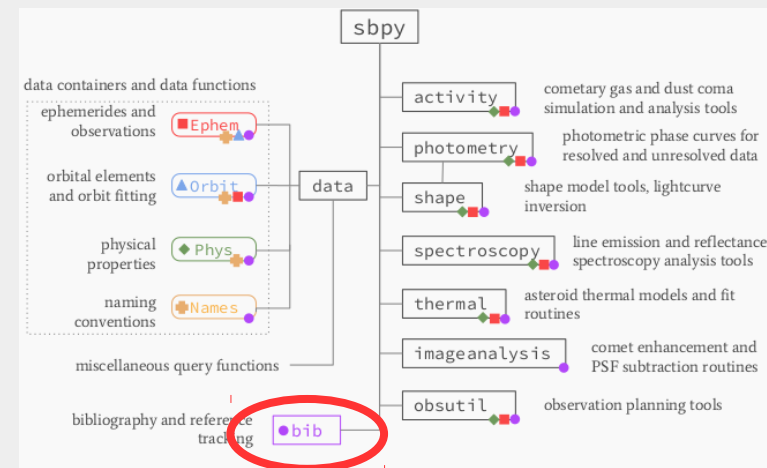
- Reference tracking throughout sbpy (basic implementation)

```
>>> from sbpy.data import Ephem
>>> from sbpy import bib
```

```
>>> with bib.Tracking():
...     eph = Ephem.from_horizons('Ceres')
```

```
>>> print(bib.to_text())
sbpy.data.Ephem.from_horizons:
  data service: Giorgini, Yeomans, Chamberlin et al. 1996,
  AAS/Division for Planetary Sciences Meeting Abstracts #28,
  25.04
```

- Generate reference lists for publications
 - clear text, BibTeX, AASTeX, Icarus, MNRAS...



sbpy – Current Status

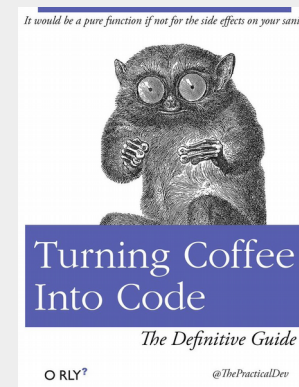
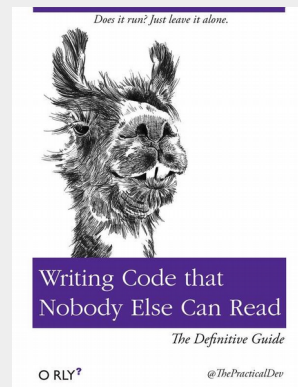
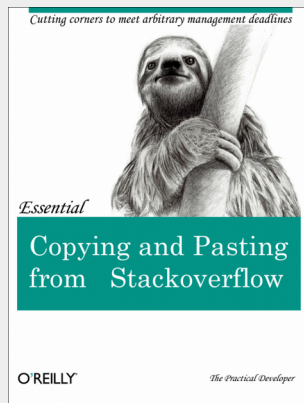
- **sbpy.data**: mostly complete
- **sbpy.spectroscopy**: some functionality implemented
- **sbpy.activity**: some functionality implemented
- **sbpy.photometry**: disk-integrated phase function models implemented
- All other modules currently only skeletons

Full functionality will be established by Summer 2021.

sbpy need your help!

- Use it, test it, break it! Then give us your **feedback!**
- What would you like to see/have in the future?
- Spread the word!
- If you use it in your work, **cite it!**

sbpy.org



sbpy is supported by NASA PDART
Grant No. 80NSSC18K0987.

