Sbpy Application Report

Qingyu Zhang

June 2021

Contents

Chapter 1

Introduction

1.1 Sbpy

Sbpy is an Astropy affiliated package for small-body planetary astronomy. It is meant to supplement functionality provided by Astropy with functions and methods that are frequently used in the context of planetary astronomy with a clear focus on asteroids and comets [1].

1.2 Module Structure

Sbpy contains several sub-modules, each of which is designed to provide certain functionalities relevant to astronomy study. The figure below is from the Sbpy document website which shows the general design of the package.

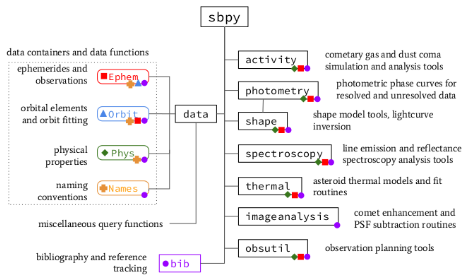

sbpy
data containers and data functions
ephemerides and observations
Ephem
orbital elements and orbit fitting
Orbit
physical properties
Phys
naming conventions
Names
data
miscellaneous query functions
bibliography and reference tracking
bib
activity
cometary gas and dust coma simulation and analysis tools
photometry
photometric phase curves for resolved and unresolved data
shape
shape model tools, lightcurve inversion
spectroscopy
line emission and reflectance spectroscopy analysis tools
thermal
asteroid thermal models and fit routines
imageanalysis
comet enhancement and PSF subtraction routines
obsutil
observation planning tools

Figure 1.1: the module structure of sbpy

Modules are shown as rectangular boxes, the colored rounded boxes are supporting modules which are the basics of this package. They are all in the data module and support many of the computations and models in this package. on the right hand side, these modules are generally applications of the data module, and each of them is designed to solve problems in a certain filed of astronomy study. We can see their Dependencies relationships from those colored symbols, where colored symbols match the modules they are using with the same colors and symbols.

In our application, we will mainly use the *activity* module to finish our computational tasks which also uses *Ephem* and *Orbit* modules as our data source.

Chapter 2

Computational Task

2.1 Task1 Orbit

This task asks us to plot the orbit and location of asteroids, we need to obtain the orbit data and then compute its location. To do that I first use the data module to get the asteroids' orbit data and ephem data, then use class Ephem's method to compute the results.

2.1.1 Work Flow

The first computational task is to compute the orbit of a planet and its position at designated time.

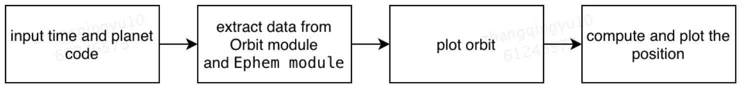

input time and planet code
extract data from Orbit module and Ephem module
plot orbit
compute and plot the position

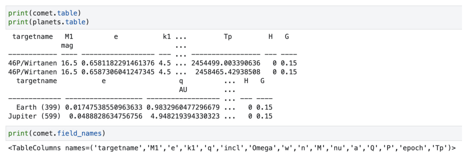
Figure 2.1: task2 workflow

2.1.2 User Input

The application needs user to input the information of the comet or planet including target-id, target-type and so on. Users can get these information through `https://ssd.jpl.nasa.gov/horizons.cgi#top`

2.1.3 Data Extraction

Take the default input as an example, we want to plot the orbit of the *46P*, *Earth* and *Jupiter*. Part of the Orbit data and filed names are shown blow.



```
print(comet.table)
print(planets.table)
```

```
 targetname   M1          e          k1 ...        Tp          H   G
             mag                        ...
------------ ---- ------------------ --- ... ---------------- --- ----
46P/Wirtanen 16.5 0.6581182291461376 4.5 ... 2454499.003390636   0 0.15
46P/Wirtanen 16.5 0.6587306041247345 4.5 ...  2458465.42938508   0 0.15
  targetname          e                    q          ... H   G
                                           AU         ...
------------- ------------------- ------------------ ... --- ----
  Earth (399) 0.01747538550963633 0.9832960477296679 ...   0 0.15
Jupiter (599)  0.0488828634756756  4.948219394330323 ...   0 0.15
```

```
print(comet.field_names)
```

```
<TableColumns names=('targetname','M1','e','k1','q','incl','Omega','w','n','M','nu','a','Q','P','epoch','Tp')>
```

Figure 2.2: Orbit Data

2.1.4 Computation and Visualization

Code below demonstrates the core of the orbit plotting

```
epochs = orb['epoch'] + np.linspace(-1, 1, 1000) * orb['P'] / 2
eph = Ephem.from_oo(orb, epochs=epochs, dynmodel='2')
ax.plot(eph['x'].value, eph['y'].value, **plot_kwargs)
```

After running the program we will get the orbit plot below. From the outside to the inside, the orbits of Jupiter, comet, and Earth respectively.

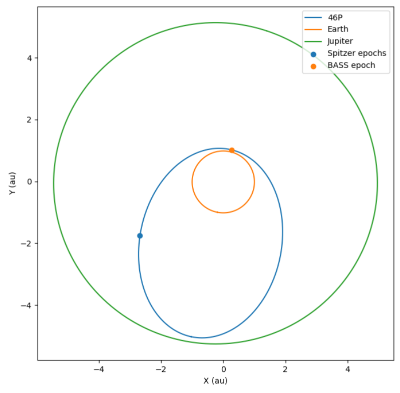


Figure 2.3: Orbit Plot

2.2 Task2 Haser Model

This task asks us to compute the distribution of coma molecules under photolysis, it's very physical and it's from one of the examples from the Sbpy tutorial, but it's easy to solve using the Haser model which has been packaged in the *activity* module, so that I can use it to compute the result straight forward.

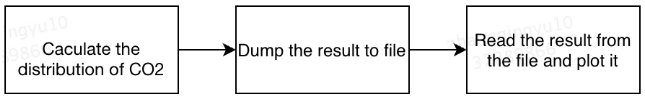

Caculate the distribution of CO2
Dump the result to file
Read the result from the file and plot it

Figure 2.4: Haser model work flow

2.2.1 Work Flow

2.2.2 Computation

Calculate the Haser model column density from 1 to 10^4 km for CO_2, produced at a rate of 10^{28}/s. Let the expansion velocity be 0.8 km/s. Let the heliocentric distance range from $1au$ to $10^{11}au$. Code blow shows the computation of the model.

```
gamma = v * tau * (rh / u.au)**2
co2 = gas.Haser(Q, v, gamma)
```

Then save the result object using pickle.

```
with open('haser_ds', 'wb') as f:
    pk.dump(res, f)
```

2.2.3 Visualization

First We read the result

```
with open('haser_ds', 'rb') as f:
    res = pk.load(f)
rho = res['rho']
data = res['data']
```

Figure blow is the plot of the distribution of CO_2

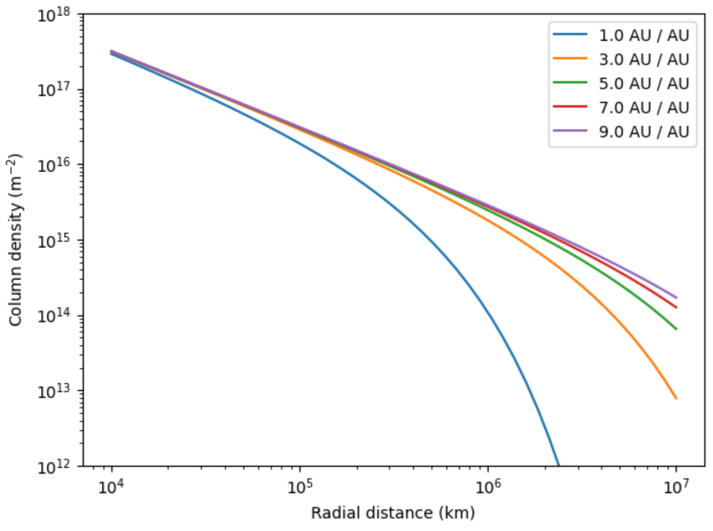


Figure 2.5: Distribution of CO_2

Chapter 3

Conclusion

Sbpy is an useful package to assist researchers to handle observation data and manipulate them. It also supply many modules other than data, which is quite difficult for me to understand their application. I used the data module and the activity module, it broadened my field of vision, I've never had a chance to touch professional Astronomy data, fresh but difficult.

Most Importantly, it helps me getting more familiar with those techniques and tools learned from this course.

Bibliography

[1] Michael Mommert, Michael S. p. Kelley, Miguel de Val-Borro, Jian-Yang Li, Giannina Guzman, Brigitta Sipőcz, Josef Ďurech, Mikael Granvik, Will Grundy, Nick Moskovitz, Antti Penttilä, and Nalin Samarasinha. sbpy: A python module for small-body planetary astronomy. *Journal of Open Source Software*, 4(38):1426, 2019.