BOLLUM: BOLometric LUMinosity code - Documentation

Created by Niko Pyykkinen

May 2024

1 Introduction

BOLLUM is an *Interactive Python Notebook* (ipynb) for creating bolometric light curves of astrophysical objects. BOLLUM is an ipynb file that can be run with Jupyter Notebook. Due to different kinds of astronomical objects and datasets, BOLLUM can be run in a few different ways.

2 Methodology

BOLLUM only works under the assumption that the spectrum of the object is close to a blackbody spectrum. The code then uses *spectral energy distribution* (SED) fitting to determine the parameters of the blackbody function at each epoch. In principle, the measured fluxes are compared to blackbody intensities at the filters' wavelengths. The code therefore systematically tries multiple different values for the free parameters R, the object's radius, and T_e the object's effective temperature, and chooses the best fit with the χ^2 method. Error margins are chosen to be the highest and lowest values with which the fit represents the sample with σ confidence. However, there are two different ways of doing this. Either the filter function can be integrated over each trial blackbody spectrum or the filter flux can be transformed to the blackbody intensity of the object at its effective wavelength.

2.1 Flux at specific wavelength

Transforming the flux to the specific intensity at the effective wavelength of the flux is done by dividing the measured flux by the effective width of the filter. The effective

width is the width of a square with the same area as the filter. Now this intensity is assigned to be the intensity at the effective wavelength of the filter, which is the weighted average of the flux of the standard star vega through the filter.

This method works best for filters the functions of which are nearly squares. *Sloan Digital Sky Survey* (SDSS) *ugriz* filters are fairly close to squares while *UBVRI* filters will have larger error with this method. Overall this method is computationally effective and allows a low step to be chosen when iterating the trial values of the free parameters.

This requires the effective wavelengths and effective widths of the filters. These can easily be acquired through *SVO filter profile service* at *http://svo2.cab.inta-csic.es/theory/fps/*[1][2].

With this method, it is possible to examine the individual fits. To do this create a folder called ims in the directory of the code and change `images=False` to `True`.

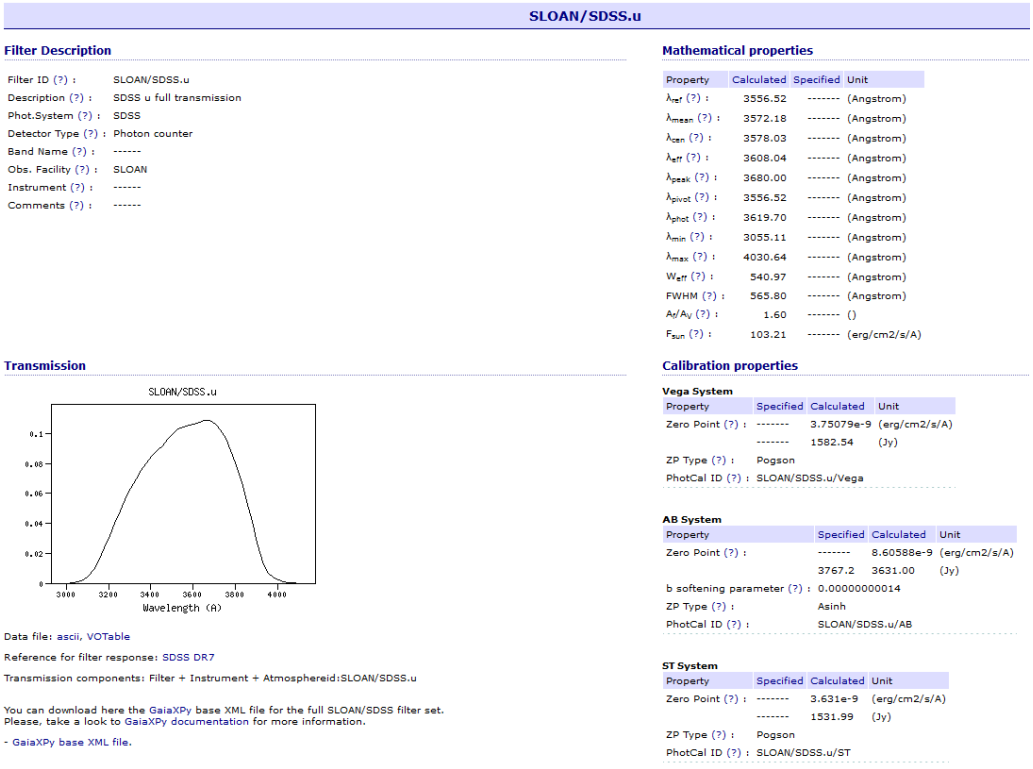

Figure 1: Example of acquiring filter parameters from SVO Filter Profile Service. These are the SDSS u-band filter parameters.

```
eff_wavelength_u=3608.04
eff_width_u=540.97
zero_point_u=3.75079e-9

eff_wavelength_g=4671.78
eff_width_g=1064.68
zero_point_g=5.45476e-9

eff_wavelength_r=6141.12
eff_width_r=1055.51
zero_point_r=2.49767e-9

eff_wavelength_i=7457.89
eff_width_i=1102.57
zero_point_i=1.38589e-9

eff_wavelength_z=8992.26
eff_width_z=1164.01
zero_point_z=8.38585e-10
```

Figure 2: Example inputting parameters into the code.

2.2 Integrating with filter functions

Here the blackbody spectrum is integrated over the filter function and the measured flux is compared with the observed flux. This is accurate and has no larger error on *UBVRI* filters as it does on *ugriz*. This method however demands much more computation time. It is not recommended for large datasets.

For this, the filter functions are needed. They as well are easily acquired through SVO. BOLLUM uses *astroquery* package to acquire the filter functions.
E.g. `function_u = SvoFps.get_transmission_data('SLOAN/SDSS.uprime_filter')` sets the variable `function_u` to contain the filter function of SDSS u filter.

3 How to run the code

To run the code you will need the light curves and relevant filter information. The effective wavelength, effective width, and zero point of the filter have to be inputted manually.

```
mjd_set=[]
for i in np.arange(-20,200,1):
    mjd_set.append(i)
```

Figure 3: Setting interpolated epochs. (Start date, end date, time step)

3.1 Setting up light curve data

There is a built-in function that will read the light curve file. This function expects a file in csv form with a header for the filter names eg. mjd,u,u_er,g,g_er,r,r_er,i,i_er. The function will also accept and correct the light curve for extinction and arbitrary time shift in case you want to display the data according to eg. the peak date of a supernova. Any epochs where there are no measurements on a band should have the value as null (Note that in csv simply no value is the null value).

3.2 Interpolating light curves

If your data has light curve measurements on different bands on different days, you will want to interpolate the data. For this change the `interpolate=False` to `True`. Note that the code does not bin measurements on the same band on the same day. To avoid losing valuable data, bin your observations on the same days before interpolating. When interpolating select the epochs manually.

3.3 Choosing free parameters

The blackbody spectrum can not be fitted into any two points perfectly when there are two free parameters. For the SED fitting to describe the blackbody spectrum minimum of three band observations are needed for each epoch. Alternatively, either the radius or the temperature of the black body can be considered a constant. To choose either of the free parameters as constant simply switch eg. `Radius_free=True` to `False`.

4 Output

The code outputs plots of each bolometric luminosity, temperature evolution and radius evolution as well as upper and lower limits for each of these errors. In addition, it will create a file called "objectname_BOLLUM.txt" with all the same information in text.

```
Day,L,L_l,L_u,T,T_l,T_u,R,R_l,R_u,khi2
0,33.57775787241735,33.37266839961737,33.80415853909146,6000,57006400,64000000000,56000000000,73000000000,1558.9293009369262
1,33.54088940113289,33.30158967645071,33.78554340240399,5500,52005800,73000000000,62000000000,87000000000,1379.586202058031
2,33.62893490655539,33.381824122866675,33.775072358894796,5300,52005700,87000000000,68000000000,89000000000,905.53605404556
3,33.56436620767538,33.17552892738495,33.92113969419784,5500,50006200,75000000000,58000000000,89000000000,268.2152713801365
4,33.56436620767538,33.31548739636136,33.80528491045658,5500,52005800,75000000000,63000000000,89000000000,821.4338641337188
5,33.58835057488523,33.58835057488523,33.58835057488523,5900,59005900,67000000000,67000000000,67000000000,9762.043638205589
6,33.57889564195458,33.49250347025087,33.67450928630967,6100,59006200,62000000000,60000000000,67000000000,14149.055283564687
```

Figure 4: Output file. Subscripts u and l are the lower and upper limits of the fit. "khi2" is the lowest χ^2 value.

In the output in Jupyter, there will be a warning if any of the epochs have either the best fit or the error reaching the lowest or highest free parameter trial value. This means that the range of the trial values needs to be increased or if the lower and upper values are already unphysical the fit at the epoch is simply poor. For SNe in particular the spectrum is similar to the blackbody spectrum at early epochs but the assumption becomes worse over time.

If you have reason to suspect the fits to be not working correctly, please enable logs by changing `logs=False` to `True`. This will print in the output the bands used in each epoch. This allows you to compare to your data and see if all the data is used in making the bolometric light curve. Depending on whether you are interpolating and whether you have one or two free parameters epoch with less than three dates might be skipped. (Please see sections 3.2 and 3.3).

5 Examples

5.1 Output images

The code saves the following images. For ease of use the plotting is done in a separate cell.

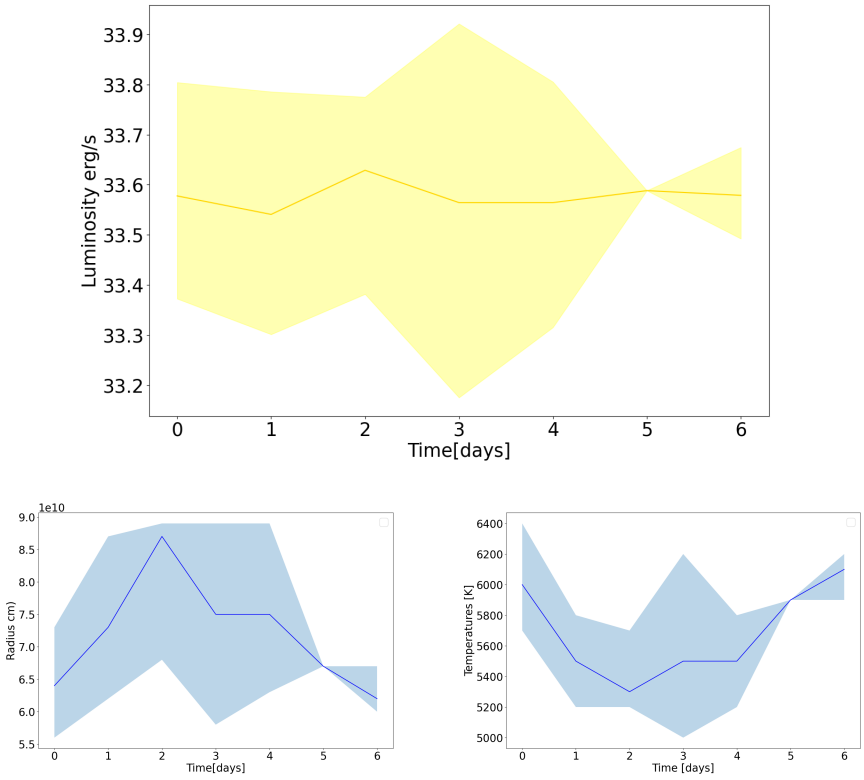

Figure 5: Top: Bolometric light curve of the sun over 5 epochs with randomly generated error. Middle: Radius evolution of the sun. Bottom Temperature evolution of the sun. The values here appear to change over time this is due to the large arbitrary error to highlight the error margins.

References

[1] C. Rodrigo, E. Solano, and A. Bayo. *SVO Filter Profile Service Version 1.0*. IVOA Working Draft 15 October 2012. Oct. 2012.

[2] C. Rodrigo and E. Solano. "The SVO Filter Profile Service". In: *XIV.0 Scientific Meeting (virtual) of the Spanish Astronomical Society*. July 2020, 182, p. 182.