Replicating A Box Plot Fitting Function In Python

Max Cooley

May 25, 2022

1 Introduction

Exoplanets are a more recent field of study within astronomy. The first exoplanet was discovered in 1992 by Aleksander Wolszczan and Dale Frail as they noticed the pulses from the pulsar PSR B1257+12 were observed to be at irregular intervals. This resulted in the discovery of two planets orbiting the pulsar with an orbital period of 67 and 98 days. There are three techniques used when attempting to find exoplanets

The first is through direct image. This, unfortunately is only possible on rare occasions. Usually with bright planets orbiting at great distances from a nearby star. The rare requirements for this method are why it is responsible for the discovery of only 50 exoplanets.

The second is through the Doppler shift or the wobble method. As a planet orbits a star there is a gravitational force on the planet due to the star and on the star due to the planet. This force can cause the star to wobble slightly. The spectrum from these stars will have a periodic velocity shift the observation of this shift is limited, as ground based observations can measure Doppler shifts greater than 3 m/sec which correspond to a planet with minimum mass of 33 times the mass of the Earth at a distance of 1 AU from a one solar mass star.

Finally exoplanets can be detected through their transits. When the orbit of a planet around its star passes in between the star and Earth it is possible to detect the reduced amount of light from the star. These dips in light will be periodic as the planet continues its orbit. This periodic dip can be measured and from that properties of the system, such as stellar mass, planet mass, and orbital period, can be determined.

Discovering the transits of exoplanets became the idea for my capstone project and I began on creating a box fitting function used to determine the period of the transit which can be used to determine details of the system, such as orbital period, planetary and stellar mass, and orbital eccentricity.

2 Process

To start I pulled data from the Transiting Exoplanet Survey Satellite (TESS) in order to test the code. The data was accessed from the Mikulski Archive Space Telescope (MAST) using the MAST Portal. The data was from the planetary system know as both WASP-86 and KELT-12. This data was selected as it allowed the results created to be verified as it had been previously measured.

Then a function was written in Python that would generate the model transits. This function would take an array for time and a value for the transit depth (T), width (dt), and center (t0). It would then create an array the length of the time array with 1 for every value. It would then loop through the length of the time array and give the value of transit depth if the position was within:

$$t0 - dt/2$$

$$t0 + dt/2$$

The function would then return the model transit.

In order to find the best fitting model we minimize the statistic

$$\mathrm{LS} = \sum_i [d_i - M_i(T, t_0, dt)]^2 \tag{1}$$

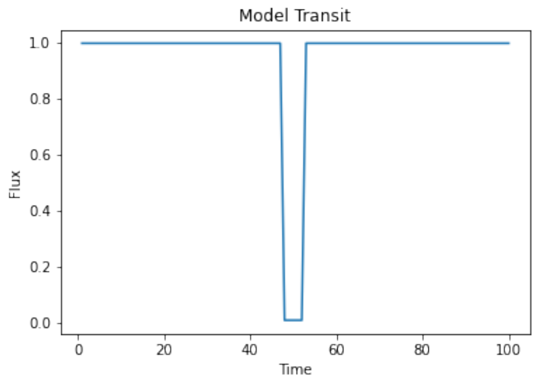
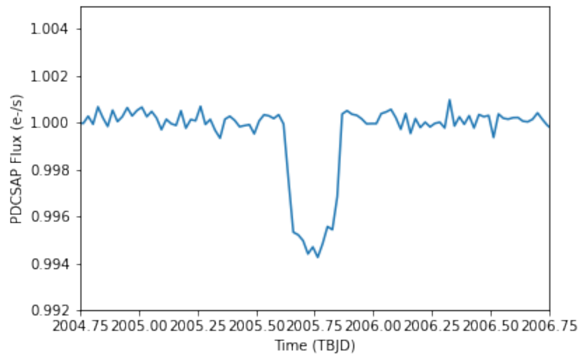


Figure 1: T=0.1 dt=5, t0=50



By minimizing this statistic would ensure that the points of the model transit are closest to that of the actual transit. This statistic was computed for a wide range of model parameters.

I also included a function that would take an array and a value and find the index of the array item that has the closest match to the given value by finding the minimum of.

$$|arrayitem - value|$$

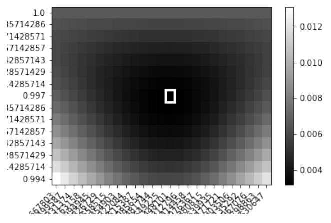
This was included to find the position of the time values to the closest whole number as the flux and time arrays from TESS data was reduced to include only one transit. The flux was also limited to a maximum value of one by dividing the array by the median value as shown below.

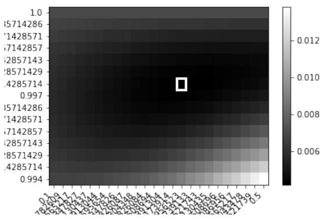
Now that it was narrowed to a single transit three arrays were created, each housing the test values for a given variable. For this data transit depth ran from 0.994 to 1 with 15 values, transit width from 0.1 to 0.5 with 24 values, and transit center 2005.576 to 2005.888 with 25 values. Then a three dimensional array of zeros with a length equal to each of the variables length. Then with nested loops a box plot was created with each combination of values. From this model the deviation from the true value and the model was taken and summed along and placed into the three dimensional array.

This array was then marginalized along each axis and displayed using the matplotlib function imshow creating the following images. Highlighted in white on these images are the minimum values that should make up a box plox that best fits the transit data. Next the data was once again marginalized over each axis and plotted with the values of the array in order to show the value that brought a minimum in the box plots.

3 Analysis

For the values of transit depth and transit center the program returned a single value minimum. However, for the transit width it returned two values: 0.2217 and 0.2391. In attempts to get the transit width to minimize to a single value the number of values in the width was increased arbitrarily


(a)


(b)

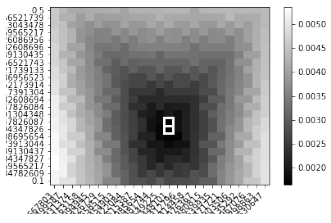
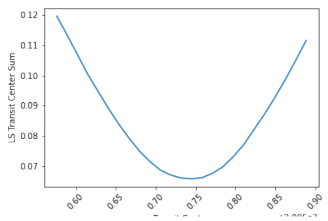
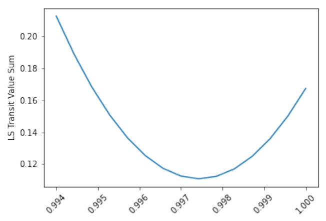

(c)

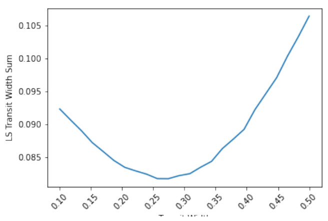
Figure 2



(a)



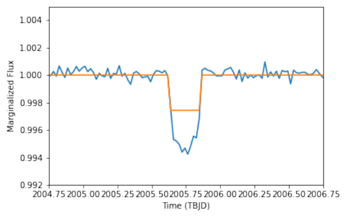
(b)



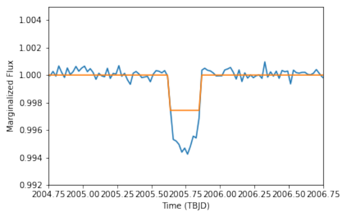
(c)

Figure 3

to 34. This however did not reduce the minimized values. The following figures display the actual transit flux and the generated box plot. The two transit widths were too small to be displayed on the same figure. It can be seen that while the transit width and transit center appear to be consistent with the actual flux, the transit depth found is not and appears about halfway up the curve. It is currently unclear why the fitting code fails to identify the proper depth of the transit.



(a)



(b)

Figure 4