Astroinformatics

Practice Grade 3

Authors:

Jose Ruben Carcamo Portillo jose.carcamo.portillo@ua.cl

Professor: NINA HERNITSCHEK

July 4, 2024

Contents

1 Objective **2**

2 Introduction **3**

3 Task 1 **4**
3.1 Light Curves 4

4 Task 2 **7**
4.1 Outliers in light curves 8

5 Task 3 **11**
5.1 Amplitude 11

6 Conclusion **12**

1 Objective

1)Plotting light curves using the libraries like matplotlib

2)Defined outliers and how to identify in light curves plots

3)Applied statistics on light curves plots.

2 Introduction

In this Practice we are going to work with TESS plotting light curves, identify outliers, and applying statistics but know is necessary to know what is TESS.

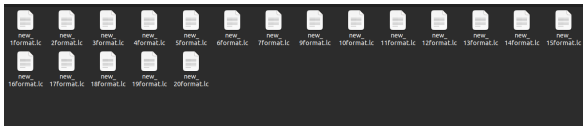
The Transiting Exoplanet Survey Satellite (TESS) is a mission designed to search for Earth-sized planets orbiting bright stars. It uses high-precision photometry to monitor over 200,000 stars continuously for two years. TESS captures data in two formats: Target Pixel Files (TPFs), which are postage stamps of individual stars, and calibrated light curves, providing detailed information on star brightness over time. Additionally, TESS records Full Frame Images (FFIs) every 30 minutes, covering a wide field-of-view enabling photometry of any object within its 24x96 degree observation range. [2]

3 Task 1

3.1 Light Curves

1) Repeat the plots from graded practice 1, but now with Python using the light curve files from practice 2. For the plots, take into account how to make them more readable.

To complete our task we'll be using the files of light curve obtained in the last practice and write python code to plot the light curves



(a) Files in lc format from the last practice

```
import pandas as pd
import matplotlib
from matplotlib import pyplot as plt
graph1 = pd.read_csv('practice3/new_6format.lc', delimiter= " ")
plt.plot(graph1['TIME'], graph1['SAP_FLUX'],color="red", lw=0.3, ls='-')
plt.xlabel('Time/JBD-2457000, days')
plt.ylabel('SAP_FlUX e/s')
plt.title('Light Curve 6')
```

(b) Script to open and plot a light curve

Figure 1: files and code

To make more readable the plotting was modify in color and width to be more friendly, the labels from x axis it's time in BJD that is Barycentric Julian Date, it refers to a timekeeping system that is used to specify the timing of events such as observations or measurements of celestial objects and in y axis is SAP_FLUX this is refer to simple aperture photometry light curve and have units of electron/seconds [4].
Plotting the 5 light curve we have this:

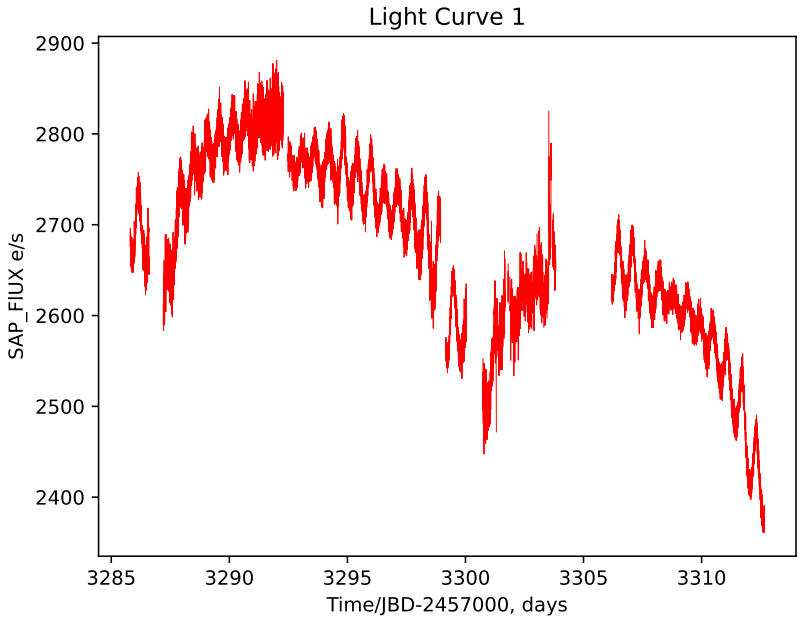


Figure 2: Light Curve #1

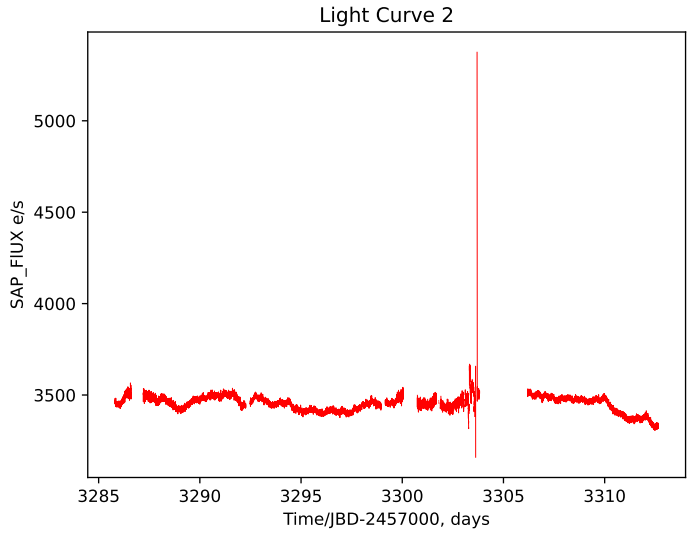


Figure 3: Light Curve #2

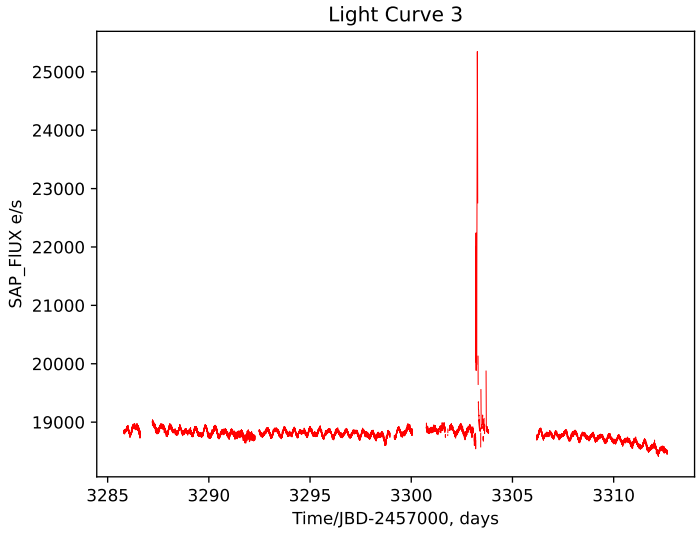


Figure 4: Light Curve #3

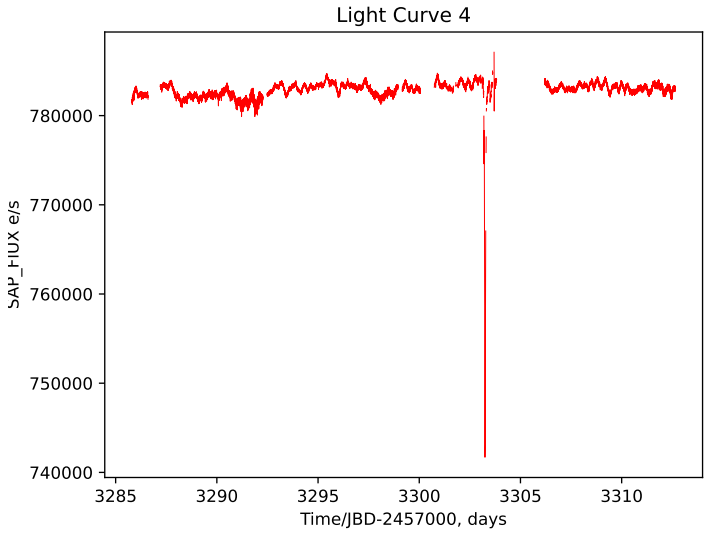


Figure 5: Light Curve #4

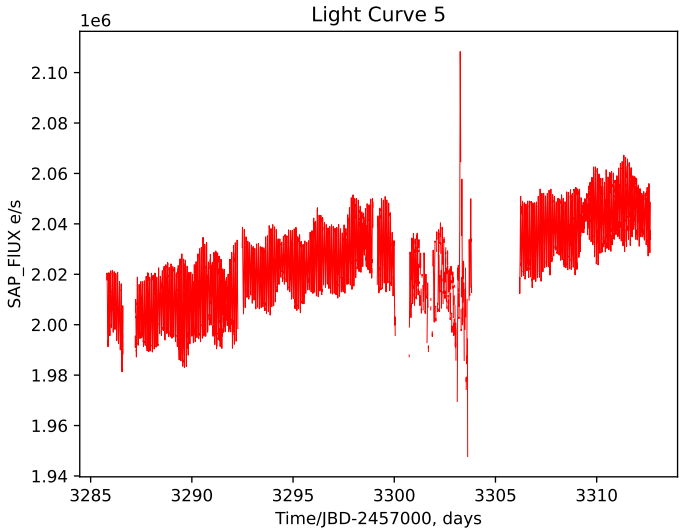


Figure 6: Light Curve #5

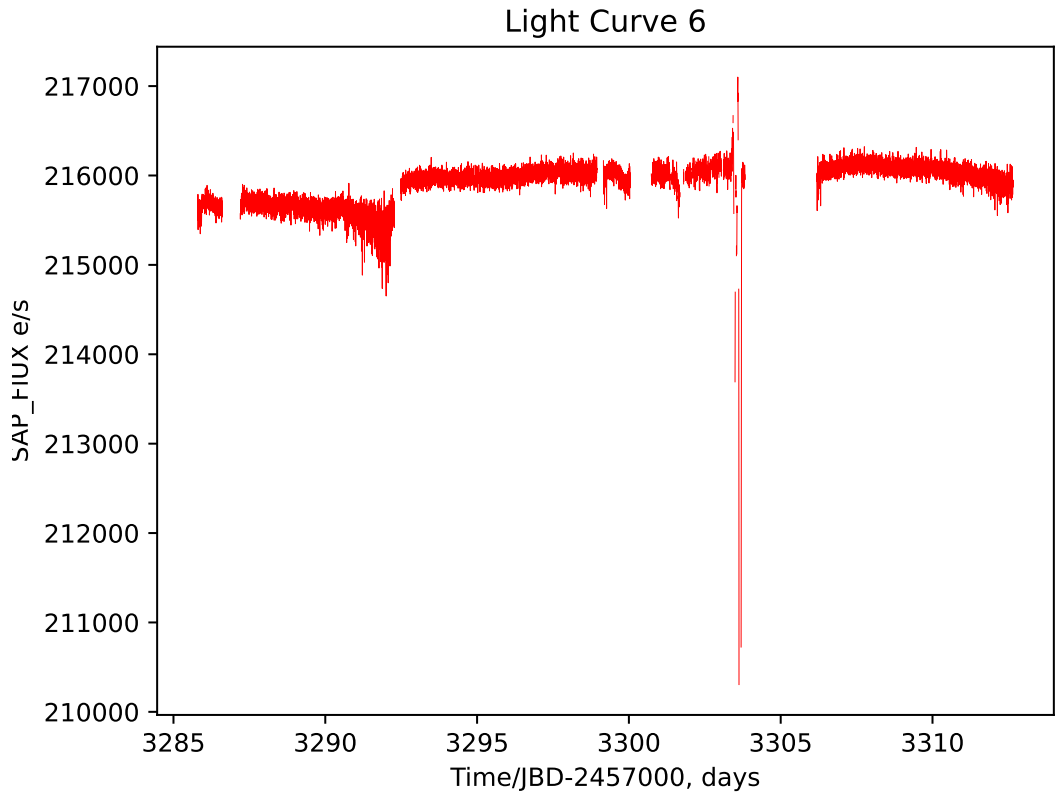


Figure 7: Light Curve #6

4 Task 2

2) When you make the plots, can you identify outliers? Highlight them. Try writing code to identify at least the most extreme outliers.

As we saw in the latter plots there are some peaks that's no fit in general with the behavior of the plots, in order to identify this outliers a python scripts will help to detect some of this outliers. To identify outliers is necessary to compare our values if is too high or low in this case i define a upper limit and lower limit to enclose all date and make a comparison for this i use a z-score method(the code for identify outliers its not mine [1]) that refer to taking values out of the ranges of 3σ, after that i plot the outliers to identify so the result was this:

```
: import pandas as pd
  import matplotlib
  import numpy as np
  from scipy import stats
  from matplotlib import pyplot as plt
  graph1 = pd.read_csv('practice3/new_6format.lc', delimiter= " ")
  plt.plot(graph1['TIME'], graph1['SAP_FLUX'],color="red", lw=0.3, ls='-')
  plt.xlabel('Time/JBD-2457000, days')
  plt.ylabel('SAP_FlUX e/s')
  plt.title('Light Curve 6')

  ul = graph1['SAP_FLUX'].mean() + 3*graph1['SAP_FLUX'].std()
  ll = graph1['SAP_FLUX'].mean() - 3*graph1['SAP_FLUX'].std()
  outliers = graph1.loc[(graph1['SAP_FLUX'] > ul) | (graph1['SAP_FLUX'] < ll)]
  plt.plot(outliers['TIME'], outliers['SAP_FLUX'],color="blue", marker='s', markersize=3, linestyle='none', label='Outliers')
  plt.legend(loc=4)
```

Figure 8: python code for outliers

4.1 Outliers in light curves

To compare both plots, the outliers are in blue to identify easily with the light curve,in the first plot there is alike sinusoidal behavior so the outliers is not good define, recall that is using the mean and deviation standard in order to discriminate the values who not fit.

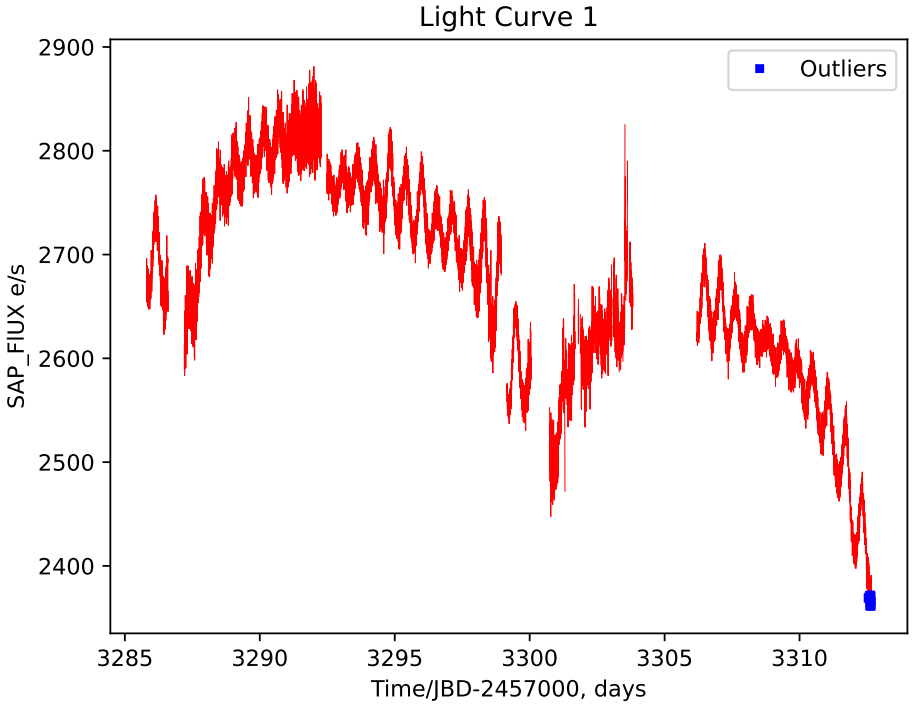


Figure 9: Light Curve #1 with outliers

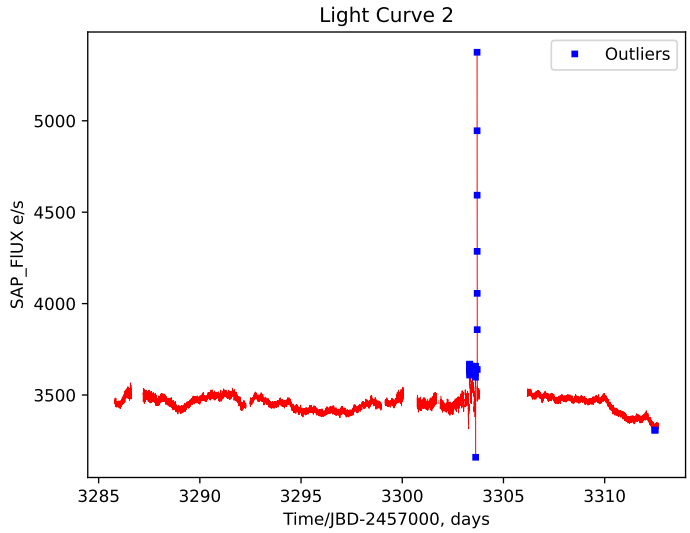


Figure 10: Light Curve #2 with outliers

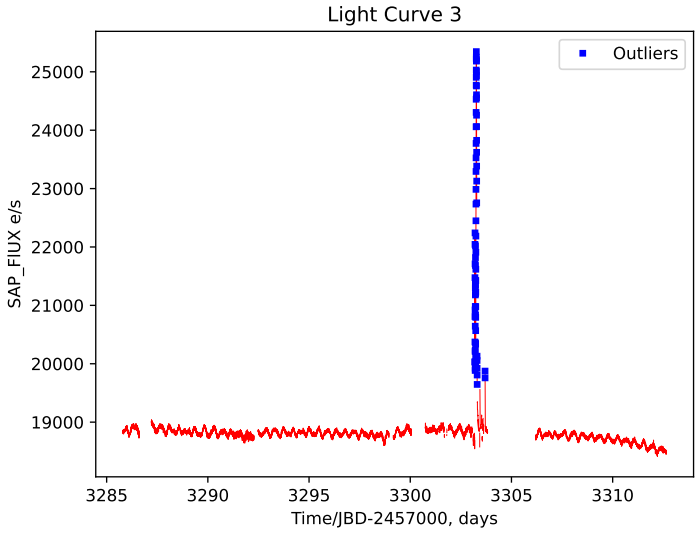


Figure 11: Light Curve #3 with outliers

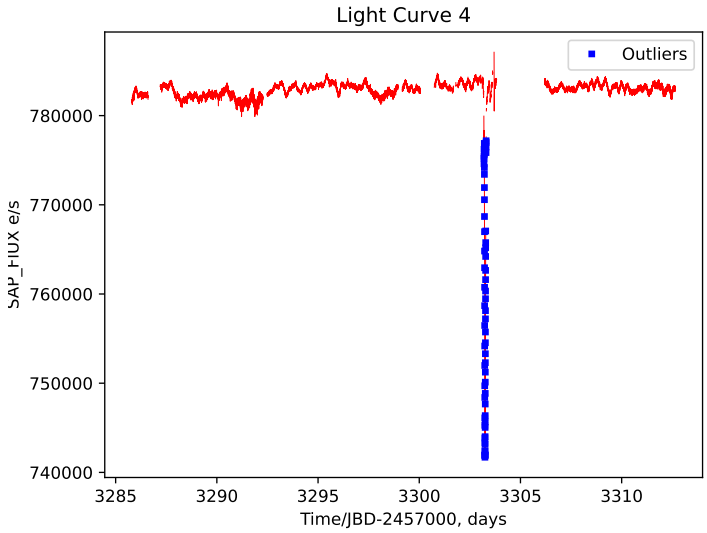


Figure 12: Light Curve #4 with outliers

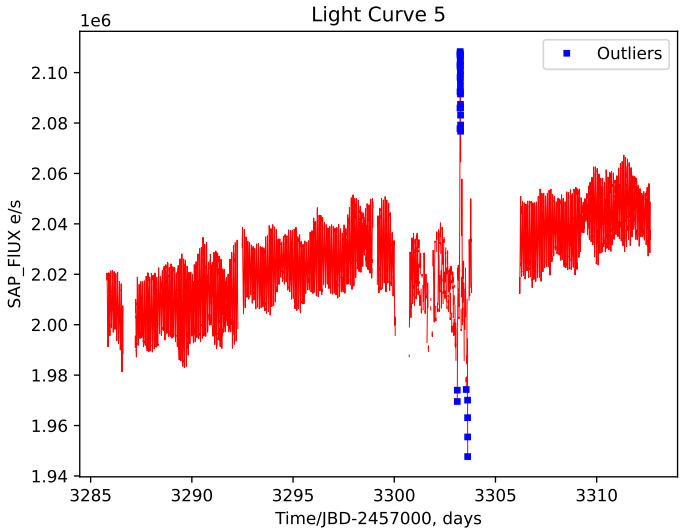


Figure 13: Light Curve #5 with outliers

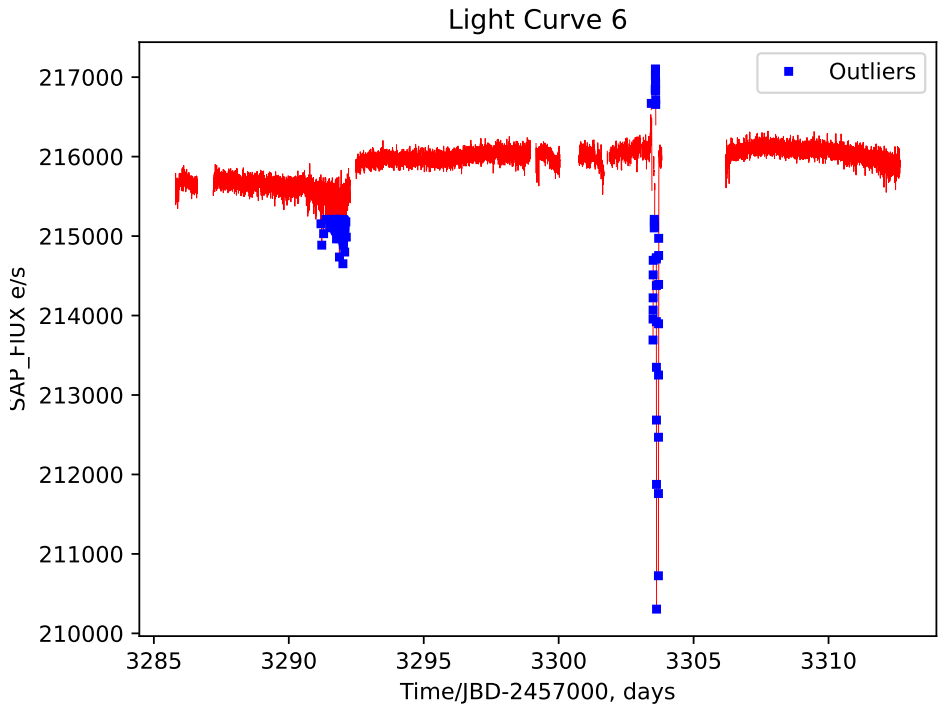


Figure 14: Light Curve #6 with outliers

5 Task 3

3)Think about basic statistics to describe light curves, such as amplitudes, and implement at least two of them.

5.1 Amplitude

The amplitude is well know as a measure of its variability, typically defined as the difference between the maximum and minimum flux values to get that is necessary to use the numpy to get those values with np.max() for max and np.min()for min [3]. Another statistic is the mean flux that computes the average flux value across all time points in the flux array. It represents the typical brightness level of the object during the observation period, and its calculate with mean_flux = np.mean(flux) [3]. In general thi values for the plots are:

```
#amplitude
amplitude = np.max(graph1['SAP_FLUX']) - np.min(graph1['SAP_FLUX'])
print(f"Amplitude of the light curve: {amplitude:.2f}")
#mean flux
mean_flux = np.mean(graph1['SAP_FLUX'])
print(f"Mean flux of the light curve: {mean_flux:.2f}")
```

Figure 15: script code for Amplitude and mean flux

Plots	Amplitude	Mean flux
new_1format.lc	520.25	2677.04
new_2format.lc	2215.26	3452.12
new_3format.lc	6936.03	18805.68
new_4format.lc	45402.64	782813.84
new_5format.lc	160640.70	2025210.46
new_6format.lc	6795.58	215909.58

The code and results are in jupyter notebook.

6 Conclusion

In this practice we learned how to plot light curves, identify outliers and mark in the plot and put statistics application to this light curves.

References

[1] `https://www.youtube.com/watch?v=Cw2IvmWRcXs`.

[2] Mikulski archive for space telescope. Tess.

[3] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[4] Douglas A. Caldwell Joseph D. Twicken. Tess science data products description document.

List of Figures

1 files and code . . . 4
2 Light Curve #1 . . . 4
3 Light Curve #2 . . . 5
4 Light Curve #3 . . . 5
5 Light Curve #4 . . . 6
6 Light Curve #5 . . . 6
7 Light Curve #6 . . . 7
8 python code for outliers . . . 8
9 Light Curve #1 with outliers . . . 8
10 Light Curve #2 with outliers . . . 9
11 Light Curve #3 with outliers . . . 9
12 Light Curve #4 with outliers . . . 10

13 Light Curve #5 with outliers 10
14 Light Curve #6 with outliers 11
15 script code for Amplitude and mean flux 11