*Exo*2: An Exploration into a Novel Characterization Method for K2 Photometry

Sameer Singh

November 2020

Contents

1 Introduction and Literature Review

An exoplanet is any planet outside of the solar system. With Aleksander Wolszczan's 1992 discovery of a triple-exoplanetary system orbiting the millisecond pulsar PSR1257 + 12 (Wolszczan Frail, 1992), a new era of astronomy began. Having previously thought the existence of exoplanets extraordinary improbable at best, astronomers avidly began to analyze pulsar timing variations in hopes of finding other such exoplanetary systems – sadly, this quest was largely doomed from the onset; astrostatistical analysis has revealed that less than one percent of all pulsars have a planetary mass companion (Martin et al. 2018), and several analyses of some long-term datasets, such as the North American Nanohertz Observatory for Gravitational Waves' 11-year 45-Millisecond Pulsar dataset, have revealed no exoplanets at all (Behrens et al. 2019). As astronomers consequently turned away from exoplanet hunting in the manner in which it was initially done, the so-called transit method – beginning with the discovery of the exoplanet HD 209458b in 1999 (Charbonneau et al. 2000) – became increasingly popular. To date, the vast majority of confirmed exoplanets have been discovered using the transit method.

In 2009, the Kepler Space Telescope was launched by NASA to collect transit-photometric data from a single field of view in the Cygnus and Lyra constellations. The mission yielded great success in its early phases, collecting in its first six weeks data from nearly 160,000 stars, which would soon thereafter be used to identify five new exoplanets – each of which changed our understanding of possible planetary properties (Borucki et al. 2010). As of September 2020, Kepler has been responsible for providing the data necessary for the discovery of 2,392 out of 4,276 known exoplanets (National Aeronautics and Space Administration, 2020). The successes of Kepler continued until 2012, when two of Kepler's reaction wheels malfunctioned, rendering the mission incapable of focusing solely on one field of view without drifting (Howell et al. 2014). As an attempt to salvage the mission, the spacecraft was repurposed as K2, collecting data while pointing near the ecliptic in its 0.5 AU orbit around the Sun (Howell et al. 2014). As of September 2020, K2, which was officially terminated in 2018, has been responsible for the discovery of 422 exoplanets (NASA, 2020).

Astronomers have long been seeking methods to characterize the vast quantity of data collected by the Kepler and K2 missions. While classification of stellar light curves as either indicative of an exoplanet or indicative of a false positive is possible to do manually, as has been done by the citizen science project "Planet Hunters," responsible for the discovery of exoplanet Kepler 64-b, this task is arduous, time-consuming, and subject to human error. To account for this error, astronomers have turned to computational methods to discover exoplanets in the K2 data.

In a watershed 2018 paper, Vanderburg & Shallue automated the process of exoplanet identification using convolutional neural networks. Convolutional neural networks – which are especially well-suited to image recognition – read in images as inputs, assign relative importance to each part of the image, and differentiate these parts of the images from one another. Standard feed-forward neural networks, on which the architecture of convolutional neural networks are built, are also able to perform image recognition tasks (as in the classic “handwritten digit recognition” case), but because convolutional neural networks are built upon a non-linear and non-fully connected foundation, they are much better at complex image recognition. In the case of distinguishing between the u-shaped dip of a transiting exoplanet and the very similar v-shaped dip of an eclipsing binary, the nuance of a convolutional neural network is indispensable. Vanderburg Shallue used a dataset of nearly 16,000 planets to generate their neural network, which trained and validated light curves generated from the original Kepler mission. Their model was able to classify exoplanets with a recall of 0.95 at a precision of 0.90, meaning that it successfully classified 95 percent of real planets as planets, and that 90 percent of its classifications were real planets (Shallue & Vanderburg, 2018). Other researchers have used random forest classifiers in place of deep neural networks, which, to the trained-eye, seems a poor approach, as the problem of exoplanet detection is not traditionally thought of as a step-wise task. However, because projects such as Planet Hunters have been quite successful – having identified four exoplanetary candidates for follow-up study by professionals (Fischer et al. 2012), and the manual classification of light curves is inherently a stepwise process, random forest classifiers, which themselves are constructed upon decision trees – stepwise structures – are a logical choice. McCauliff et al. created the “Autovetter Project” based on random forest classifiers to automatically classify the shape of light curves as either belonging to exoplanets or false positives. Running their algorithm on Kepler Threshold-crossing events, they were able to distinguish between systematic noise, eclipsing binaries, and exoplanet candidate signatures with an error rate of just 5.85 percent, reduced further to 2.81 percent when distinctions between astrophysical false positives and systematic noise were reduced (McCauliff et al. 2015). Similar work was done by Mislis et al. in 2016, achieving a success rate of greater than 95 percent (Mislis et al. 2016).

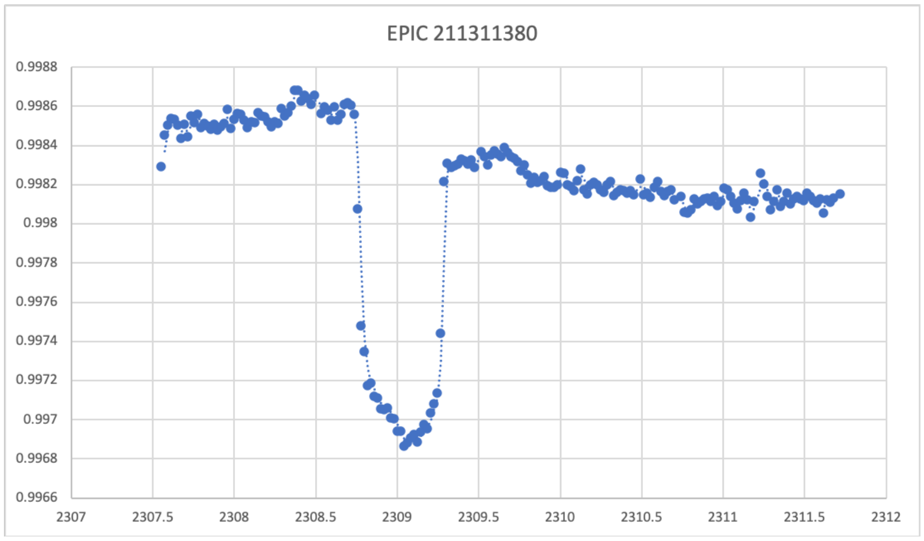
2 Rationale and Methods

While machine learning methods as discussed above are extremely precise, they are both difficult to understand and necessarily require large amounts of computer storage. For example, the convolutional neural network developed by Shallue and Vanderburg requires a minimum of 90 gigabytes of storage [1]. When running their download script on a standard configuration 2019 MacBook Air, and directing the downloads to an external solid state drive, I found that the FITS (Flexible Image Transport System) files that were required to construct their program – named AstroNet – took 213 gigabytes of storage. Indeed, FITS files are the ”industry standard” in astronomy, and neural networks such as those constructed by the aforementioned authors require huge volumes of input – or, more formally ”training” data. These requirements are prohibitive for citizen science, as many citizen scientists do not have the financial means to

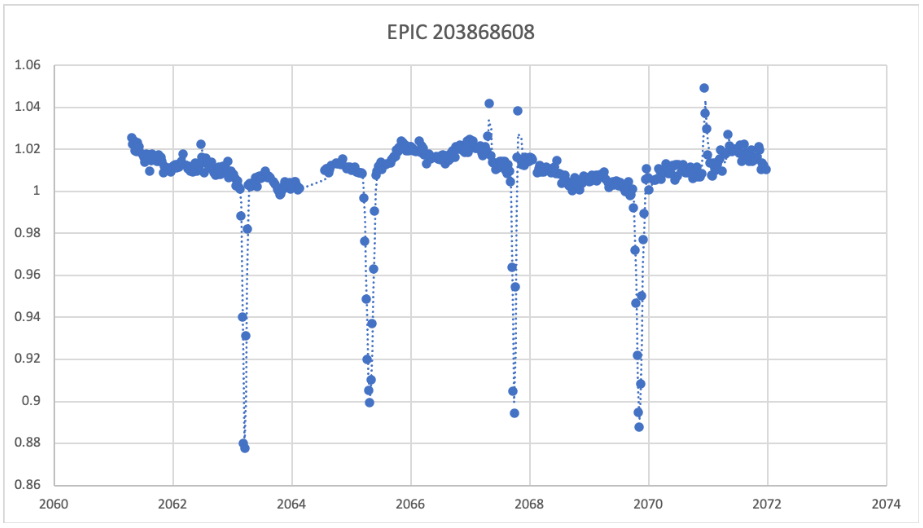
[1] https://github.com/google-research/exoplanet-ml/tree/master/exoplanet-ml/astronet

buy more storage space than is provided on their laptops. Moreover, advanced understanding of computer science and statistics is required for operation of machine learning algorithms, further restricting access to citizen scientists. When machine learning algorithms require extensive debugging as did AstroNet, those who are not well-versed in advanced programming content such as the writing of Unix shell scripts and the use of Tensorflow for ML are at a significant disadvantage. As a response to the aforementioned problems, I propose a new exoplanet identification program, significantly more accessible in both usability and storage requirements: Exo2. The development of Exo2 made use of Lightkurve, a Python package for Kepler and TESS data analysis.[2] This research also made use of keplersplinev2, an unpublished Python module for spline fitting for time-series astronomy.[3]

The fundamental premise of the Exo2 model is the following: exoplanet transits produce u-shaped transits, whereas eclipsing binaries – which are binary star systems orbiting on a plane in our line of sight, and thus regularly crossing one another, and perhaps the most common source of false positives in exoplanet detection – produce v-shaped transits.



[2]https://github.com/KeplerGO/lightkurve
[3]https://github.com/avanderburg/keplersplinev2

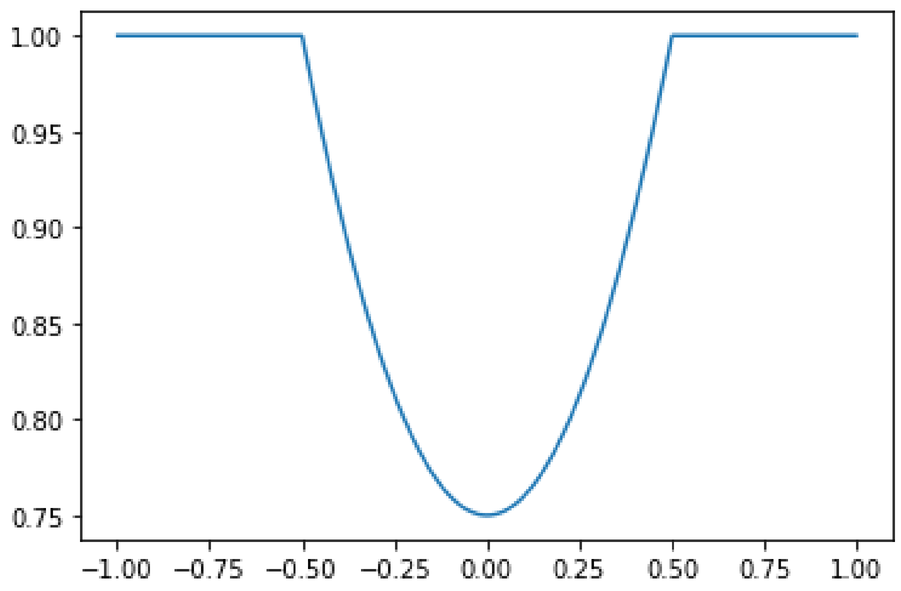


The above are raw plots of transits visible in exoplanet and eclipsing binary light curves, respectively. EPIC (Ecliptic Plane Input Catalog) IDs are unique identifiers for stars observed by the Kepler spacecraft. The y-axis data represents flux in units of electrons-per-second. The flux is contained within the optimal aperture pixels collected by the Kepler/K2 spacecraft. The x-axis data represents time in units of Baryocentric Julian Days. Plots were generated using Microsoft Excel and data points were fitted with a moving average trend line to emphasize transits.

The mathematical notion of the derivative is key to understanding the distinction between u-shaped and v-shaped transits, and is indeed key to programming a computer to recognize the difference between the two. In a general sense, the derivative is equal to the rate of change of a function at a given point: for functions involving curvature, i.e. those that are nonlinear, the first derivative is most often given as another function. Building upon this notion, the second derivative is the rate of change of the rate of change of a function at a given point. If we model a normalized ideal exoplanet light curve as a mathematical function, where t is time and *f(t)* is flux with respect to time, it might be defined as follows:

$$f(t) = \begin{cases} 1 & \text{if } -1 < t < -0.5 \\ x^2 + 0.75 & \text{if } -0.5 < t < 0.5 \\ 1 & \text{if } 0.5 < t < 1.0 \end{cases} \tag{1}$$

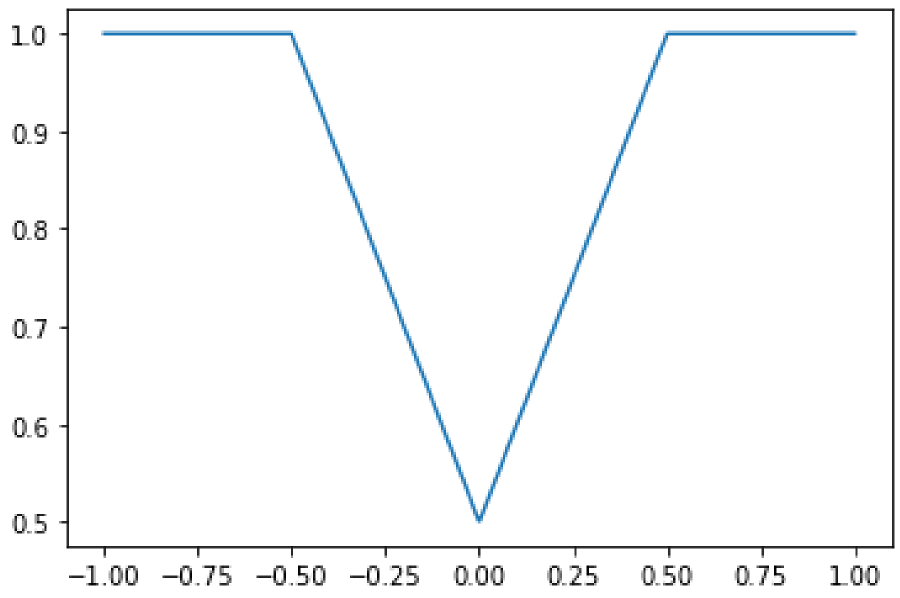
This function is pictured below and was graphed using Python's matplotlib library.



Conversely, if we model a normalized ideal eclipsing binary (false positive) light curve as a mathematical function, where t is time and *g(t)* is flux with respect to time, it might be defined as follows:

$$g(t) = \begin{cases} 1 & \text{if } -1 < t < -0.5 \\ -t + 0.5 & \text{if } -0.5 < t < 0 \\ t + 0.5 & \text{if } 0 < t < 0.5 \\ 1 & \text{if } 0.5 < t < 1.0 \end{cases} \tag{2}$$

This function is pictured below.



With this idealization of the geometries of the light curves, we can begin our analyses of the rates of change. Denoting the second derivative of the functions as $f''(t)$ and $g''(t)$, we obtain the following values:

$$f''(t) = \begin{cases} 0 & \text{if } -1 < t < -0.5 \\ 2 & \text{if } -0.5 < t < 0.5 \\ 0 & \text{if } 0.5 < t < 1.0 \end{cases} \tag{3}$$

$$g''(t) = \begin{cases} 0 & \text{if } -1 < t < -0.5 \\ 0 & \text{if } -0.5 < t < 0 \\ 0 & \text{if } 0 < t < 0.5 \\ 0 & \text{if } 0.5 < t < 1.0 \end{cases} \tag{4}$$

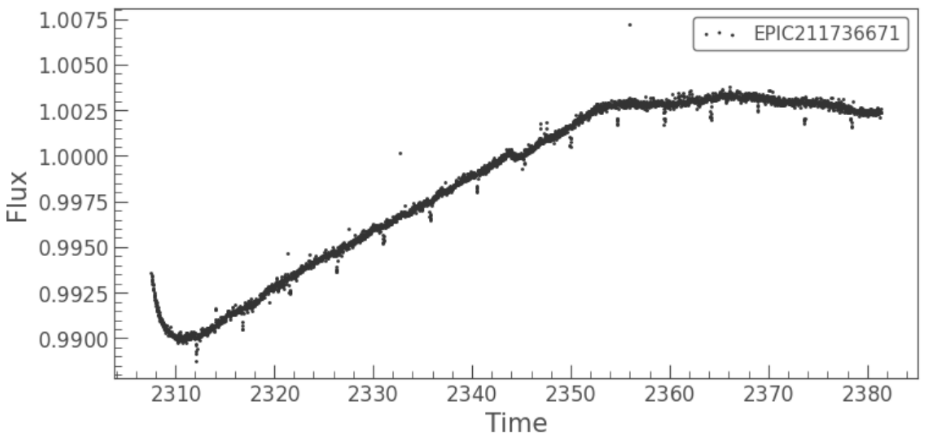
It is on the theoretical results (3) and (4) that Exo2 is based: in theory, we expect the graphs of the rate of change of the rate of change for exoplanet light curves to be non-zero at the transit, where it will equal a constant value; false positive graphs, however, will be zero for the entirety of their domain. Of course, because these are only mathematical approximations, and there is much noise in K2 data, we do not expect our theoretical results to be in perfect concordance with the experimental results. Indeed, statistical noise in the K2 data greatly renders most of the data far from appearing like the mathematical models established in (1) and (2). According to Howell et al. 2014, raw K2 aperture photometry is anywhere between 3 and 4 times less precise than that produced by the original Kepler mission. Vanderburg and Johnson further suggested in 2018 that the pointing jitter of K2 (a result of the missing components discussed in Section 1) has led to its photometry being dominated by noise rather than true signals, unlike its predecessor Kepler mission. Consequently, transits are scarcely defined as clearly as they were in the graphs of EPIC 211311380 and EPIC 203868608. Additionally, the transit depth of most exoplanets, which is most commonly approximated as

$$\text{Depth} = (R_p/R_s)^2 \tag{5}$$

where R_p is the planetary radius, and R_s is the stellar radius, typically has a value of less than 0.01, or 1 percent – the 1 percent threshold exists for planets on the order of Jupiter or Saturn (Carter & Winn 2010).

The first step in addressing these issues was to select an appropriate source of data. A major systematic source of noise for K2 Photometry is thruster fires, occurring when the spacecraft is accelerating. Andrew Vanderburg maintains a database of K2 light curves corrected for thruster fires as described in Vanderburg & Johnson 2014.[4] Crucially, all these light curves are available as text (txt) files, which are orders of magnitude smaller than FITS files, and which – unlike FITS files – do not required specialized applications to view. Using *wget* commands in shell script, light curves were downloaded for all 19 K2 Campaigns (each corresponding to an observational period of about 80 days).

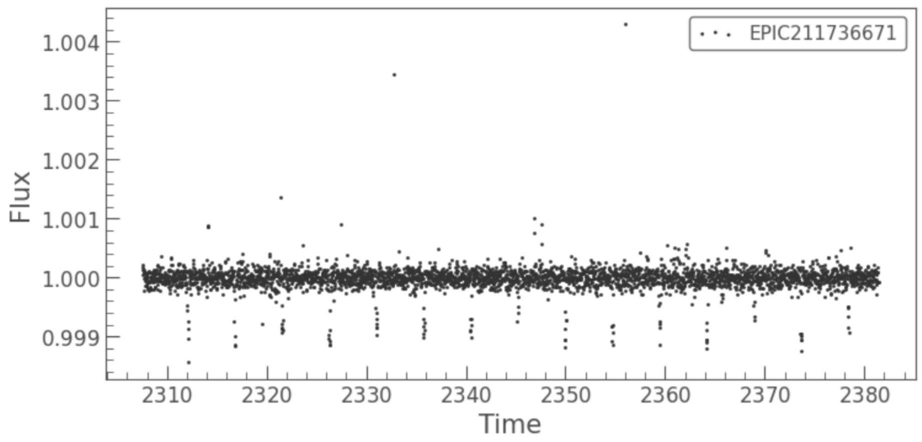
Even though the light curves were free of thruster fire errors, the majority of the statistical noise remained. Pictured below is the light curve for confirmed exoplanet EPIC 211736671.



Two primary problems emerge when looking at at the light curve: the first being that there is an extensive long-term trend in the data, and the second being that there is extensive low frequency variability in the data as well. To begin the process of addressing the latter, use was made of the open-source

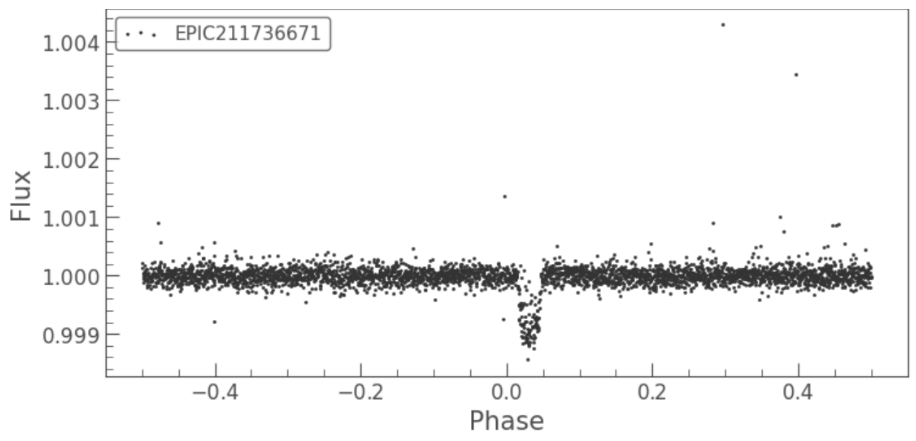
[4]https://www.cfa.harvard.edu/avanderb/k2.html

Python library astropy[5] to clip from the data all points that had a value 3σ greater than the mean flux. Although outlier removal traditionally entails the removal of both points smaller than $\overline{x} - 3\sigma$ and larger than $\overline{x} + 3\sigma$, because removing low outliers could remove crucial transit features. Next, lightkurve's methods for implementing the Savitzky-Golay filter were used to remove long-term trends from the data as well as low frequency variability therefrom, i.e, to "smoothen" the data. After removing outliers and applying the Savitzky-Golay filter, the light curve of EPIC 211736671 appears as follows:

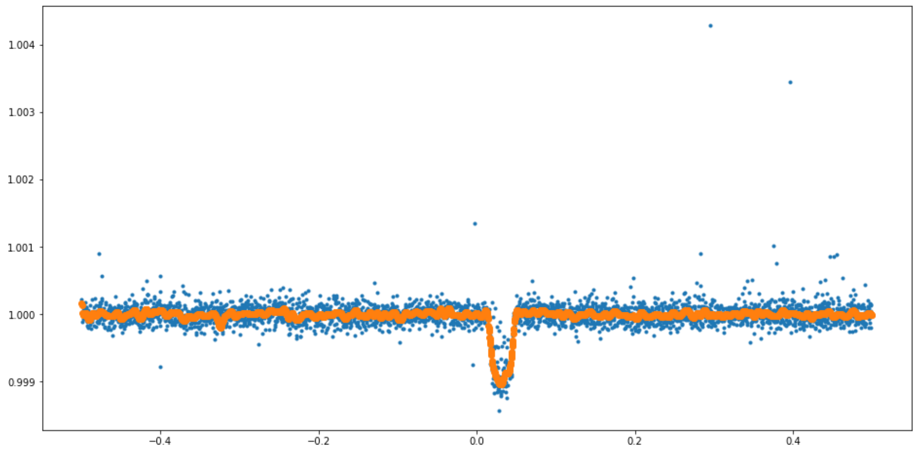


To complete the process of preparing the data for numerical analysis, a technique known as "phase-folding" was implemented. Because several transits are generally present in a light curve, one can more clearly define the transit by aligning flux values with respect to the transit maximum. Phase folding allows data to span one orbital period by "folding" several transits atop one another – hence the name. To phase-fold, one needs to know the period of the transiting phenomenon in question. As there is no *a priori* way to decude this parameter, probabilistic methods are used. Kovacs et al. 2002 proposed a Box Least Squares algorithm for determining periodicity, and it has become widely accepted in the astronomy community. The algorithm essentially model transits as tophats ("boxes") and uses this information to generate a plot of periods vs their likelihoods of fitting to the data – such plots are known as "periodograms." Using lightkurve's implementation of BLS allowed for phase folding by permitting the passage the outputs of the periodogram as arguments for period in phase folding. After phase folding the light curve of EPIC 211736671, it appears as follows:

[5]https://github.com/astropy/astropy



After removing outliers from the data, smoothening it, and phase folding it, splines were fitted to the data. Splines are special piecewise defined functions that make use of several polynomials to model noisy data. For this particular purpose, splines were fitted using the iPython module keplersplinev2, specially optimized for modelling Kepler-mission light curves. After splines were fitted to the data, the splines were written to csv files using the Python library pandas[6], at which point it became possible to run numerical analyses of rates of change (and rates of change thereof).



Above is pictured the phase folded light curve for EPIC 211736671, with the spline fit overlayed in orange .

For every pair of tuples ($time_1$, $Flux_1$) and ($time_2$, $Flux_2$) generated from the text files, and then subsequently cleaned, phase-folded, and fitted with a spline, the rate of change m and rate of change of rate of change s are given by the following approximations of the first and second derivatives at a given point,

[6]https://github.com/pandas-dev/pandas

respectively. Because the distances between any two given points are incredibly small, these approximations hold generally true:

$$m = \frac{Flux_2 - Flux_1}{time_2 - time_1} \tag{6}$$

$$s = \frac{m_2 - m_1}{time_2 - time_1} \tag{7}$$

3 Discussion

4 Conclusions and Future Research Directions

5 Acknowledgments