T.C.

AKDENIZ UNIVERSITY

MODELLING OF EXOPLANETARY TRANSITS AND ITS LIGHT CURVES

Kaya Han TAŞ
20183405003

INSTITUTE OF NATURAL AND APPLIED SCIENCES

SPACE SCIENCES AND TECHNOLOGIES

DEPARTMENT

GRADUATION THESIS

JUNE 2022

ANTALYA

CONTENTS

ABSTRACT

For exoplanetary transits, a Python code was written in order to analyze their light curves. In this code, the aim is to make a fit to the light curve of the transit and find the parameters of the exoplanet's orbit and limb-darkening coefficients accurately with their errors included. For the fitting of the light curve, an "Eclipse Function" was used, which is also given in this paper. The code also uses the Quadratic Limb Darkening Law for analysis.

To get the best fit possible, the Least Squares Method was used. By getting the best fit possible, the code is able to give us the parameters we want to analyze with accuracy. The code has been tested on Kepler-12b, Kepler-61b, and Kepler-86b systems. The results of the code were also compared to the values of the parameters in the literature.

1. INTRODUCTION

One of the methods we use in order to detect an exoplanet is the analysis of the light curves of the exoplanet's "host" star. In the case of an exoplanetary transit, the exoplanet passes in front of its host star, causing its brightness to dip. We call that dip a "minimum" or a "minima" on the light curve. Because of the limb-darkening effect on the stars, the moment an exoplanet eclipses the center of its host star is when the brightness of the star dips the most.

Transit events also happen on "Eclipsing Binary Stars" as well. So, first of all, we need to understand the difference between an exoplanetary transit and a binary star transit. When a star passes in front of another star, because of the eclipsing star having its own brightness as well, the minima would be shallow. But when an exoplanet passes in front of a star, because of the planet's own brightness being close to zero, the minima would be deeper.

Also, another difference is that in eclipsing binary stars when there isn't a transit happening, we see both the primary star's, which is the brighter component in the binary system, and the secondary star's brightness on the light curve. So, when the primary star or the secondary star gets eclipsed, there will be two different minimas seen for both of the events. Meanwhile, on the exoplanetary transits, since the planet doesn't have its own brightness, we will only see one minima on the light curve.

In this paper, the aim is to make a fit to a host star's light curve. For this first of all, we need to select a limb darkening law to work with. There are two limb darkening laws that are commonly used, Linear Limb Darkening Law and Quadratic Limb Darkening Law. This brings up the question, "*What is a limb darkening law?*".

The Limb Darkening Law is simply the difference between the light coming from the center and the limb (or the edge) of the star. In our line of sight, the light coming from the star's center would be more since it comes straight towards us. On the other hand, due to the light coming from the star's limbs being in different directions than our line of sight, the light loss is inevitable. This is important, especially in the analysis of the light curves since it affects the minima shapes.

Every star has its own limb darkening constants that rely on the star's gravity ($\mathbf{log\,g}$), effective temperature ($\boldsymbol{T}_{\mathbf{eff}}$), microturbulence ($\boldsymbol{v}_{\mathbf{turb}}$) and metallicity ($[\mathbf{Fe/H}]$). These constants can be found at the tables given in Claret (2000). In this paper an additional Python code was written for the limb darkening constants. The code finds the limb darkening constants automatically by giving the parameters of the star we discussed above. This code will also be explained in the **Appendix** part of this paper.

2. FUNDAMENTALS OF LIGHT CURVE ANALYSIS

To work with exoplanetary transits, there are two different solutions we can work with: analytical and numerical. In this paper we will be focusing on the analytical solution of the transits. At any given $\boldsymbol{t}$ **moment**, the total luminosity coming from a star is given by $\boldsymbol{L(t) = L_s(t) + L_p(t) - [\alpha(t) L_s(t)]}$. Here $\boldsymbol{L_s(t)}$ is the star's luminosity at that moment, $\boldsymbol{L_p(t)}$ is the planet's luminosity at that moment and $\boldsymbol{\alpha(t)}$ is the light loss due to the planet eclipsing its host star.

When analyzing the light curves, we normalize the total luminosity to 1, that is $\boldsymbol{L_s + L_p = 1}$. In this case, the total luminosity at any given phase θ would be equal to $\boldsymbol{L(\theta) = 1 - [\alpha(t) L_s(t)]}$. From here, we can write the total luminosity when the planet gets eclipsed by its host star as $\boldsymbol{L(\theta) = 1 - L_p = L_s}$.

$\boldsymbol{\alpha(t)}$, that is the light loss due to the eclipse, is also known as the "**Eclipse Function**". To model the light curve of a given host star we have to calculate its eclipse function for every phase. This will be discussed further in the upcoming sections.

We also discussed the Limb Darkening Law in the **Introduction** section. For the analysis in this paper, the Quadratic Limb Darkening Law is being used. The general limb darkening formula is given below, with $\boldsymbol{I(1)}$ being the intensity emerging normally to the surface. In the quadratic case, the Limb Darkening Law would be equal to $\boldsymbol{I(\mu) = I(1) * [1 - u_a(1-\mu) - u_b(1-\mu)^2]}$ with $\boldsymbol{\mu}$ being the cosine of the angle between the line of sight and the upcoming light's direction.

$$\boxed{\boldsymbol{I(\mu) = I(1) * \left[1 - \sum_{n=1}^{N} u_n(1-\mu^n)\right]}}$$

General Limb Darkening Law

In the quadratic law $\boldsymbol{N = 2}$ and there are two different limb darkening constants $\boldsymbol{u_1}$ **and** $\boldsymbol{u_2}$. We can obtain these Quadratic Limb Darkening Constants by using the formulas $\boldsymbol{u_1 = u_a + 2u_b}$ and $\boldsymbol{u_2 = -u_b}$. In here $\boldsymbol{u_a}$ **and** $\boldsymbol{u_b}$ are the constants we get from the tables we discussed in the **Introduction** section. This part is crucial when it comes to calculating the eclipse function. The connection between these constants and the eclipse function will be discussed further in the upcoming section.

3. THE ECLIPSE FUNCTION AND THE MATH BEHIND THE FITTING

In this section, we'll go over the math behind the Python Code, such as how the fitting was done, which calculations and formulas are required, which parameters must be calculated and what our input data must obtain in order to analyze the light curve of a given host star.

First of all, our input data must contain **observation time (days)** and **flux (normalized)** values. Simply, we need the light curve data in order to make a fitting. The observation times can vary. For example, one can observe at time 300 while one can observe at 600. This won't be such a problem since we convert observation times to **phase** by the formula $\boldsymbol{\theta = (T - T_0)/P}$. Now this evokes the question of "*what parameters are needed in order to make the fitting possible?*".

There are **9 parameters** needed as an input to the code to make the fitting possible. These are the parameters that we calculate the fit with and find accurately by using the least squares method to find the best fit. These parameters include the exoplanet's orbital parameters **inferior conjunction (T_0), period (P), eccentricity (e), argument of periapsis (ω), inclination (i)**, Host Star's **limb darkening coefficients (u_a, u_b)** and finally the **fractional radius of the exoplanet (r_p)** and the **star (r_s)**. All of these parameters are needed in order to calculate the eclipse function.

The Eclipse Function is given by the formula below. Here $\boldsymbol{C_n}$ is a constant that relies on the $\boldsymbol{u_a}$ **and** $\boldsymbol{u_b}$ constants and $\boldsymbol{\alpha_n}$ is the **Fractional Alpha/Eclipse Function**. Since the Eclipse Function denotes the light loss in the transit phase, on a normalized light curve, total light at any phase can be given by $\boldsymbol{L(\theta) = 1 - \alpha(\theta)}$.

$$\boxed{\alpha = \sum_{n=0}^{N} (C_n * \alpha_n)}$$

The Eclipse Function

Since we are working with the Quadratic Limb Darkening Law $\boldsymbol{N = 2}$, this means that we need to calculate $\boldsymbol{\alpha_0, \alpha_1, \alpha_2}$ and $\boldsymbol{C_0, C_1, C_2}$ values **for every phase** in order to make the $\boldsymbol{L(\theta) = 1 - \alpha(\theta)}$ **fitting to our light curve** possible. Now we have to discuss how these two parameters are calculated.

Calculating the $\boldsymbol{C_n}$ constants is quite simple when the **Limb Darkening Coefficients u_a, u_b** are given. We know how to calculate $\boldsymbol{u_1}$ **and** $\boldsymbol{u_2}$ so by using the formulas below, one can easily calculate $\boldsymbol{C_0, C_1, C_2}$ constants. Note that these constants do not change with respect to phase. A General Formula for these constants can be found in the **Appendix**.

$$\boxed{C_0 = \frac{3 - 3u_1}{3 - u_1}} \quad \boxed{C_1 = \frac{3u_1}{3 - u_1}} \quad \boxed{C_2 = \frac{6 - u_2}{6 - 2u_1 - 3u_2}}$$

Calculation of C_n constants in the Eclipse Function

However, the calculation of fractional alpha functions is a bit more complicated due to its reliance on mathematical functions. The Fractional Alpha Function formula can be found down below.

$$\boxed{\alpha_h^0 = b^2 * (1 - c^2)^{(\nu+1)} * \Gamma(\nu) * \sum_{n=0}^{\infty} \frac{n! * (\nu + 2n + 2)}{(n+1) * \Gamma(\nu + n + 1)} * \left[R_n^{1,\nu}(a)\right]^2 * R_n^{(1+\nu,0)}(c^2)}$$

The Fractional Alpha/Eclipse Function

In here $\boldsymbol{\nu = h + 2/2}$ and $\boldsymbol{\Gamma(n) = (n-1)!}$. In Mathematics, $\boldsymbol{\Gamma(n)}$ is known as the **Gamma Function**. Recall that to make a fit to the light curve, $\boldsymbol{L(\theta) = 1 - \alpha(\theta)}$ needs to be calculated for every given time in our input data. This means that for every given time (or phase) $\boldsymbol{\alpha_0, \alpha_1, \alpha_2}$ fractional alpha functions need to be calculated!

In the formula above, there are some parameters we haven't discussed yet, $\boldsymbol{a}$**,** $\boldsymbol{b}$ **and c**. The $\boldsymbol{a}$ and $\boldsymbol{b}$ parameters can be easily calculated with the given **fractional radius of the exoplanet (**$\boldsymbol{r_p}$**)** and the **star (**$\boldsymbol{r_s}$**)**. The formulas to calculate these two parameters can be found below.

$$\boxed{a = \frac{r_s}{r_s + r_p}} \quad \boxed{b = \frac{r_p}{r_s + r_p} = 1 - a}$$

Calculations of a and b parameters

The calculation of the **c parameter** is a bit more complicated since it relies on the **apparent distance between the exoplanet's center and the star's center; that is, the parameter** $\boldsymbol{\delta}$**!** Of course, this parameter also depends on the **observation time/phase** since the apparent distance between the star and the exoplanet's center changes overtime**.**

The calculation of the $\boldsymbol{\delta}$ requires some orbital mechanics knowledge. The formulas needed are given in the **Appendix**. In order to find the $\boldsymbol{\delta}$ **parameter** for every given time (or phase), we need to calculate the **Mean Anomaly (**$\boldsymbol{M_n}$**), Eccentric Anomaly (**$\boldsymbol{E_n}$**), True Anomaly (**$\boldsymbol{\nu_n}$**)** and **Distance between the components of the system (**$\boldsymbol{\rho}$**)** for every given time (or phase) respectively.

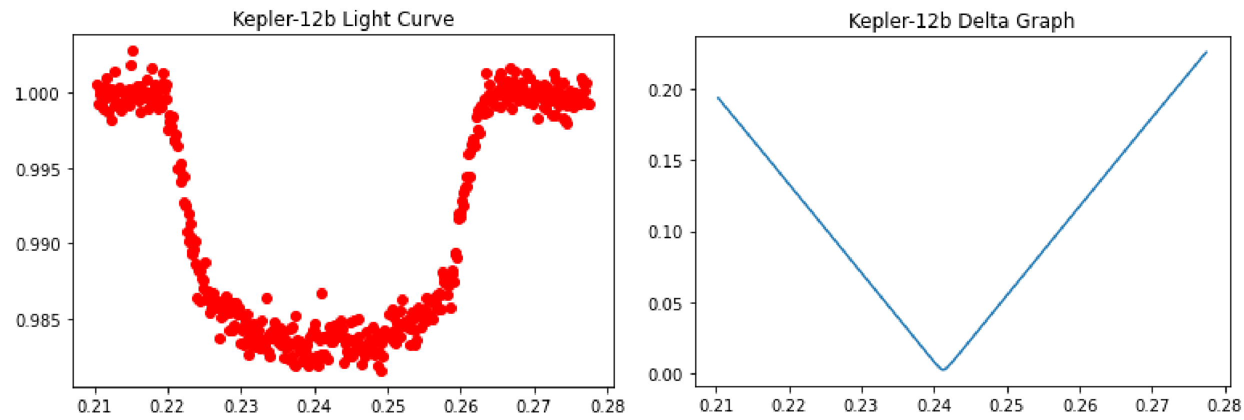
The calculation of $\boldsymbol{\delta}$ **parameter** changes depending on the **eccentricity (**$\boldsymbol{e}$**)** of the planet's orbit. If the orbit is circular $\boldsymbol{e = 0}$ and $\boldsymbol{\delta = \sqrt{(\sin^2\theta * \sin^2 i) + \cos^2 i}}$**.**

If the orbit is elliptical $\boldsymbol{0 < e < 1}$ and $\boldsymbol{\delta = \rho\sqrt{(\sin^2(\nu(\theta) - \nu(0)) * \sin^2 i) + \cos^2 i}}$**.** Note that in an elliptical orbit, true anomaly (ν_n) and Distance between the components of the system (ρ) needs to be calculated for each phase. Here $\boldsymbol{\nu(0) = 90 - \omega}$**.**

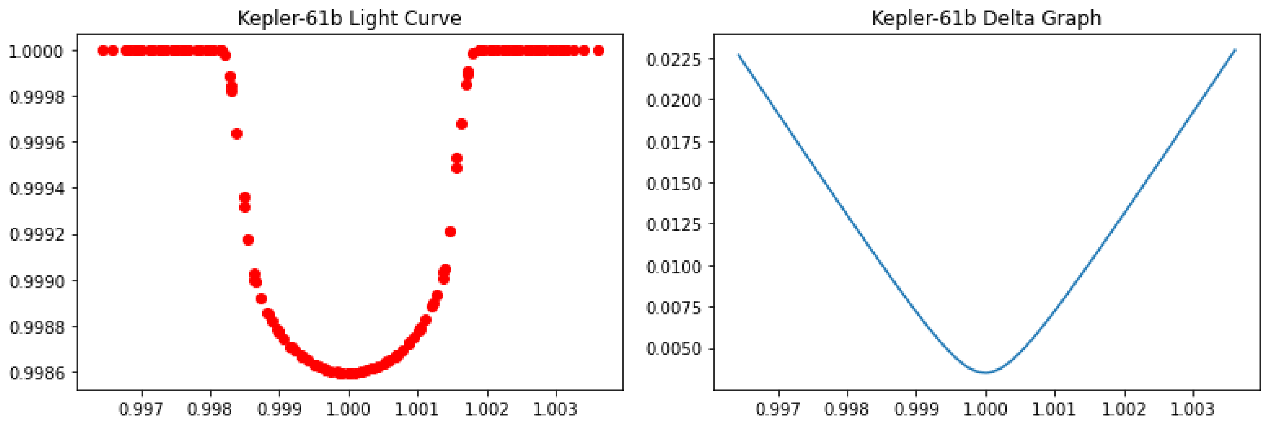
After calculating $\boldsymbol{\delta}$ **parameters** we can also calculate the **c parameter** for every given time with the formula given on the next page. Note that $c > 1$ means there are no eclipses happening. So, if $c > 1$, we can take it as $\boldsymbol{c = 1}$ since they both indicate that there are no eclipses happening. From here, we can understand that if there is an eclipse/transit, at some point $\boldsymbol{\delta}$ **and** $\boldsymbol{c}$ **have to be (or close to) zero.** An example from the Kepler-12b and Kepler-61b systems is also given on the next page.

$$\boxed{c_i = \frac{\delta_i}{r_s + r_p}}$$

Calculation of c parameter



Kepler-12b's Light Curve and the corresponding δ values



Kepler-61b's Light Curve and the corresponding δ values

Now that we know these parameters, the first part of *The Fractional Alpha/Eclipse Function* can easily be calculated for each observation time/phase. Now we have to understand the equation in the sum symbol.

In the Python code, the limits of the Sum Symbol are taken as $\boldsymbol{n = 0 \rightarrow n = 80}$ instead of the $n = 0 \rightarrow n = \infty$ limits. The upper limit of this symbol can be bigger, but for the calculation to be fast, the limit was determined as 80.

The first part of the equation inside the Sum Symbol can be calculated easily for each n. But as for the $\left[\boldsymbol{R_n^{1,\nu}(a)}\right]^{\mathbf{2}}$ **and** $\boldsymbol{R_n^{(1+\nu,0)}}(\boldsymbol{c^2})$ we need to understand a mathematical polynomial known as the **Jacobi Polynomials**. Jacobi Polynomials in general can be given as shown below.

$$\boxed{\boldsymbol{R_n^{\alpha,\beta}(x)}}$$

A Jacobi Polynomial

As it can be seen, in the first Jacobi Polynomial $\boldsymbol{\alpha = 1, \beta = \nu}$ **and** $\boldsymbol{x = a}$**.** As for the second Jacobi Polynomial $\boldsymbol{\alpha = 1 + \nu, \beta = 0}$ **and** $\boldsymbol{x = c^2}$**.** For each n in the Sum Symbol limits, these Jacobi Polynomials need to be calculated.

With **Jacobi Polynomials,** $\boldsymbol{a}$**,** $\boldsymbol{b}$**, and c** now understood, we can calculate *The Fractional Alpha/Eclipse Function* for each given phase.

We now fully understand how to calculate $\boldsymbol{\alpha_0, \alpha_1, \alpha_2}$ **for each phase** and $\boldsymbol{C_0, C_1, C_2}$. From here, using *The Eclipse Function* formula, we can calculate $\boldsymbol{1 - \alpha(\theta)}$ **which is the flux at the given phase**. The $\boldsymbol{1 - \alpha(\theta)}$ values we find here are what we call our "**Light Curve Fit**". From here the light curve can be fitted.

4. THE BEST FIT AND PYTHON CODE RESULTS

In this section, the results obtained from our Python code for Kepler-12b, Kepler-61b, and Kepler-86b will be discussed. The comparisons for our results will be made as well.

Before that, we have to discuss how the Python code works. Like we discussed in **The Eclipse Function and The Math Behind the Fitting** section, the input file has to have **observation time (days)** and **flux (normalized)**. A .txt file is recommended for the input file.

Next are the input parameters, which are embedded in the code. The parameters **inferior conjunction (T_0), period (P), eccentricity (e), argument of periapsis (ω), inclination (i)**, **limb darkening coefficients (u_a, u_b)** and **fractional radius of the exoplanet and the star (r_p, r_s)** need to be written by the user to the corresponding parameters defined in the code.

Afterwards, the user can also select **which parameters may vary** while the code tries to find the best fit. This part is also embedded in the code. The user has to write 1 or 0 if they want the corresponding parameter to vary or not, respectively. It is recommended to let the parameters vary one by one. Of course, the user can leave more parameters to vary, e.g., r_p and r_s at the same time, but letting too many parameters vary at the same time may result in huge errors in the resulting parameters.

To obtain the best fit, it is recommended to leave one or two parameters to vary and obtain the best result for those parameters with their error. Then write those results to the corresponding parameters defined in the code and repeat this with every parameter.

For example, let us assume that $T_0 = 662.55200$. Using the code, it is found that the best fit result for the parameter is $T_0 = 662.552419 \pm 0.00021615$. Then the user can write this result to the T_0 parameter defined in the code and make it stop varying. With this, the best T_0 value is obtained. The same should be done to other parameters as well. In the end, it depends on the user to find the best fit results possible.

Finally, if the input parameters are too off from the real results, the fit will not be obtained. Again, for example, let us assume that $T_0 = 662.55200$. If the user writes $T_0 = 665.55200$, the fit won't be obtained. So, the input values have to be close to the original values. This code only helps it obtain the result more accurately with the help of least squares method.

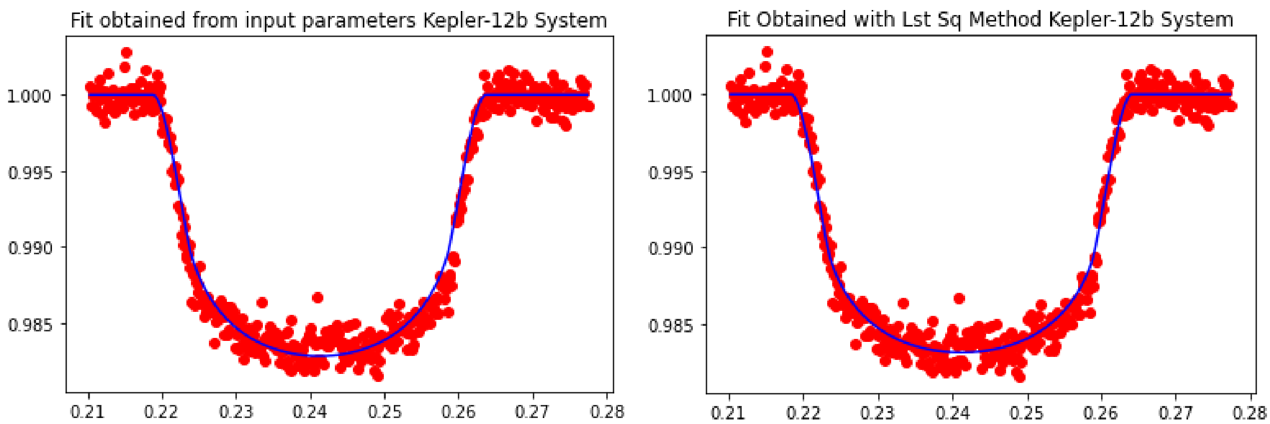
For each system, first we will check the difference between the input parameters and the resulting parameters from the code. Second, we will compare the fit obtained by the input parameters and the best fit found by least squares method. Third, we will compare the best fit parameters obtained from our code with different parameter values given in different publications.

Note that the **period (P)** values are well obtained from light curves. Because of this, it is preferred to keep the period constant instead of letting it vary. Also, if a parameter's error is too much, it is recommended to first obtain other parameters' best fit results and then go back to that parameter. For example, **inclination (i)** is preferred to be checked last on Kepler-12b's analysis.

4.1 Kepler-12b System

Kepler-12b System Parameters		
Parameter Name	**Input Parameter**	**Resulting Parameter**
Inferior Conjunction (T_0) [days]	662.55200	662.552419 $\pm$ 2.1615e-04
Period (P) [days]	4.43796	4.43773635 $\pm$ 8.0724e-04
Eccentricity (e)	0.010	0.01000090 $\pm$ 1.4260e-04
Argument of Periapsis (ω) [Radians]	3.1765	3.17650001 $\pm$ 2.8512e-04
Inclination (i) [Radians]	1.568	1.55819473 $\pm$ 0.00250990
Limb Darkening Coefficient (u_a)	0.2848	0.26552580 $\pm$ 0.00979264
Limb Darkening Coefficient (u_b)	0.3153	0.30088610 $\pm$ 0.00710165
Fractional Radius of the Exoplanet (r_p)	0.01582	0.01614167 $\pm$ 4.2611e-05
Fractional Radius of the Star (r_s)	0.12470	0.12748124 $\pm$ 3.2262e-04

Table of the Input Parameters and the resulting parameters from the Python code



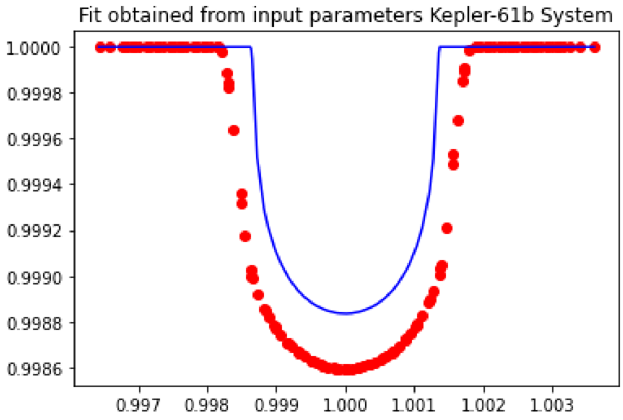
The difference between the fit obtained with input parameters and the best fit obtained with least squares method

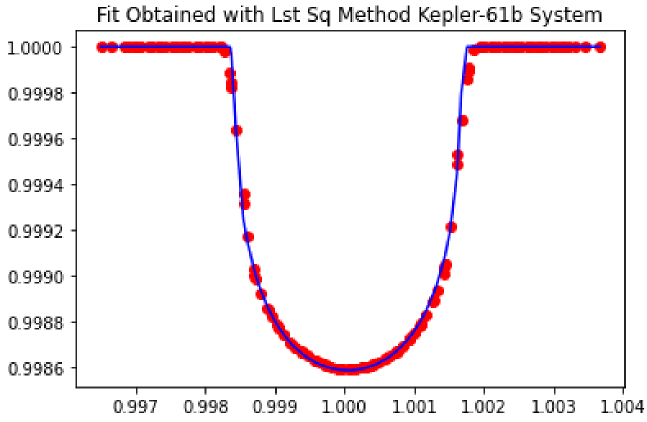
Kepler-12b System Parameters Comparison			
Parameter Name	**Fortney, et al.**	**Esteves, et al.**	**Parameter values obtained from the Python code**
Inferior Conjunction (T_0) [days]	5004.00835 $\pm$ 0.00002	171.00915 $\pm$ 0.000010	662.552419 $\pm$ 2.1615e-04
Period (P) [days]	4.4379637 $\pm$ 0.0000002	4.4379629 $\pm$ 0.0000006	4.43773635 $\pm$ 8.0724e-04
Eccentricity (e)	<0.01	---	0.01000090 $\pm$ 1.4260e-04
Argument of Periapsis (ω) [Radians]	3.1764992 $\pm$ 1.701696	---	3.17650001 $\pm$ 2.8512e-04
Inclination (i) [Radians]	1.54915 $\pm$ 0.0013962	1.549782 $\pm$ 0.015359	1.55819473 $\pm$ 0.00250990
Limb Darkening Coefficient (u_a)	0.274 $\pm$ 0.006	---	0.26552580 $\pm$ 0.00979264
Limb Darkening Coefficient (u_b)	0.367 $\pm$ 0.003	---	0.30088610 $\pm$ 0.00710165
Fractional Radius of the Exoplanet (r_p)	---	0.0148 $\pm$ 1.85497e-8	0.01614167 $\pm$ 4.2611e-05
Fractional Radius of the Star (r_s)	---	0.125 $\pm$ 2.182317e-04	0.12748124 $\pm$ 3.2262e-04
Citation	Fortney, et al., 2011, Discovery and Atmospheric Characterization of Giant Planet Kepler-12B: An Inflated Radius Outlier	Lisa J. Esteves, Ernst J. W. De Mooij & Ray Jayawardhana, 2015, Changing Phases of Alien Worlds: Probing Atmospheres of KEPLER Planets with High – Precision Photometry	---

4.2 Kepler-61b System

Kepler-61b System Parameters		
Parameter Name	**Input Parameter**	**Resulting Parameter**
Inferior Conjunction (T_0) [days]	151.185	151.181354 $\pm$ 1.6297e-04
Period (P) [days]	59.8780538	59.8780922 $\pm$ 3.2563e-05
Eccentricity (e)	0.25	0.24986015 $\pm$ 0.00166070
Argument of Periapsis (ω) [Radians]	1.570796	1.57031160 $\pm$ 2.9983e-05
Inclination (i) [Radians]	1.5673057	1.56686872 $\pm$ 1.1761e-04
Limb Darkening Coefficient (u_a)	0.3249	0.33197495 $\pm$ 0.00926068
Limb Darkening Coefficient (u_b)	0.2831	0.29036772 $\pm$ 0.00611540
Fractional Radius of the Exoplanet (r_p)	3.64348785 85e-4	4.9041e-04 $\pm$ 1.0156e-06
Fractional Radius of the Star (r_s)	0.0110375275	0.01353467 $\pm$ 2.2276e-05

Table of the Input Parameters and the resulting parameters from the Python code





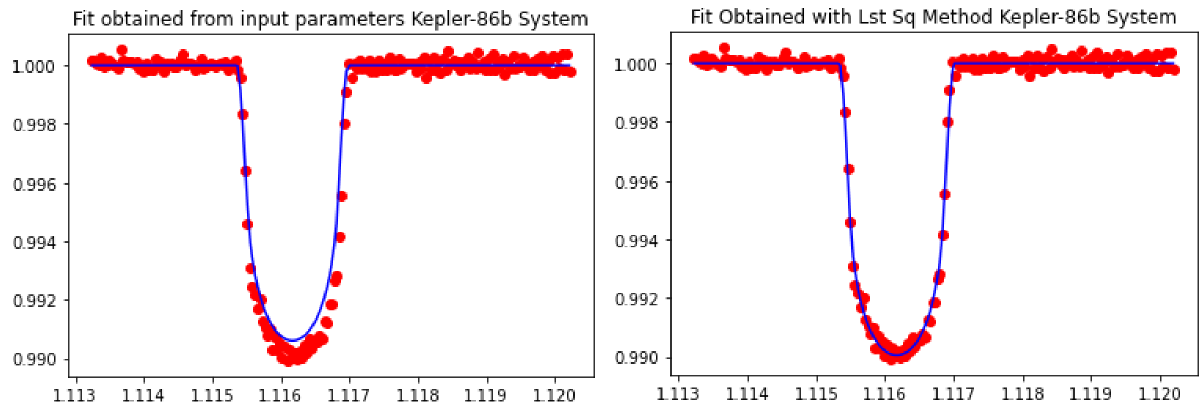
The difference between the fit obtained with input parameters and the best fit obtained with least squares method

Kepler-61b System Parameters Comparison				
Parameter Name	**Ballard, et al.**	**Parameter values obtained from the Python code (Solution 1)**	**Q1-Q17 DR25 KOI Table (Candidate Solution)**	**Parameter values obtained from the Python code (Solution 2)**
Inferior Conjunction (T_0) [days]	4984.1880 $\pm$ 0.0029	151.181354 $\pm$ 1.6297e-04	4984.185 $\pm$ 0.0021	151.184708 $\pm$ 1.5722e-04
Period (P) [days]	59.87756 $\pm$ 0.00020	59.8780922 $\pm$ 3.2563e-05	59.877824 $\pm$ 0.000162	59.8780363 $\pm$ 3.3699e-05
Eccentricity (e)	*<0.25*	*0.24986015* $\pm$ *0.00166070*	*0*	*0.00191250* $\pm$ *0.00200796*
Argument of Periapsis (ω) [Radians]	1.57031160 $\pm$	1.57031160 $\pm$ 2.9983e-05	---	1.57085879 $\pm$ 1.6593e-05
Inclination (i) [Radians]	>1.5673056	1.56686872 $\pm$ 1.1761e-04	1.568178	1.56729791 $\pm$ 4.6440e-05
Limb Darkening Coefficient (u_a)	---	*0.33197495* $\pm$ *0.00926068*	---	*0.21488204* $\pm$ *0.01006065*
Limb Darkening Coefficient (u_b)	---	*0.29036772* $\pm$ *0.00611540*	---	*0.35137056* $\pm$ *0.00597692*
Fractional Radius of the Exoplanet (r_p)	3.6434878585e-04 $\pm$ 3.5418e-07	4.9041e-04 $\pm$ 1.0156e-06	3.3983e-04 $\pm$ 2.1676e-07	4.0311e-04 $\pm$ 8.1827e-07
Fractional Radius of the Star (r_s)	0.0110375275 $\pm$ 4.1668889e-04	0.01353467 $\pm$ 2.2276e-05	0.01016 $\pm$ 2.1676e-04	0.01078104 $\pm$ 1.7463e-05
Citation	Ballard, et al., 2018, Exoplanet Characterization by Proxy: A Transiting 2.15 $R_{\oplus}$ Planet Near the Habitable Zone of the Late K Dwarf Kepler-61	---	Nasa Exoplanet Archive, Kepler-61b Planetary Parameters, Candidate Solutions	---

4.3 Kepler-86b System

Kepler-86b System Parameters		
Parameter Name	**Input Parameter**	**Resulting Parameter**
Inferior Conjunction (T_0) [days]	330.2450	330.250469 $\pm$ 4.8399e-04
Period (P) [days]	282.5255	282.525764 $\pm$ 6.0806e-04
Eccentricity (e)	0.41	0.40999999 $\pm$ 6.0980e-06
Argument of Periapsis (ω) [Radians]	0.060039	0.06003929 $\pm$ 1.4814e-05
Inclination (i) [Radians]	1.567829	1.56783950 $\pm$ 2.8654e-05
Limb Darkening Coefficient (u_a)	0.31	0.28355355 $\pm$ 0.01227994
Limb Darkening Coefficient (u_b)	0.30	0.29407333 $\pm$ 0.00644231
Fractional Radius of the Exoplanet (r_p)	0.0005435	5.7374e-04 $\pm$ 1.3372e-06
Fractional Radius of the Star (r_s)	0.005659	0.00578722 $\pm$ 1.2371e-05

Table of the Input Parameters and the resulting parameters from the Python code



The difference between the fit obtained with input parameters and the best fit obtained with least squares method

Kepler-86b System Parameters Comparison			
Parameter Name	**Wang, et al.**	**Q1-Q17 DR25 KOI Table (Candidate Solution)**	**Parameter values obtained from the Python code**
Inferior Conjunction (T_0) [days]	5196.070900	5196.0720 ± 0.00026	$330.250469 \pm$ 4.8399e-04
Period (P) [days]	282.5255 ± 0.0010	$282.5253558 \pm 0.0001219$	$282.525764 \pm$ 6.0806e-04
Eccentricity (e)	0.41 ± 0.29	0	$0.40999999 \pm$ 6.0980e-06
Argument of Periapsis (ω) [Radians]	0.060039 ± 0.470017	---	$0.06003929 \pm$ 1.4814e-05
Inclination (i) [Radians]	1.567829 ± 0.000523	1.569923	$1.56783950 \pm$ 2.8654e-05
Limb Darkening Coefficient (u_a)	---	---	$0.28355355 \pm 0.01227994$
Limb Darkening Coefficient (u_b)	---	---	$0.29407333 \pm 0.00644231$
Fractional Radius of the Exoplanet (r_p)	5.20655e-04 $\pm$ 8.83965e-08	4.3922e-04 $\pm$ 3.4974e-8	5.7374e-04 $\pm$ 1.3372e-06
Fractional Radius of the Star (r_s)	$0.0056593 \pm$ 2.94655e-04	4.7968e-03 $\pm$ 1.7487e-4	$0.00578722 \pm$ 1.2371e-05
Citation	Wang, et al., 2013, Planet Hunters. V. A Confirmed Jupiter-size Planet in the Habitable Zone and 42 Planet Candidates from the Kepler Archive Data	Nasa Exoplanet Archive, Kepler-86b Planetary Parameters, Candidate Solutions	---

5. DISCUSSION

It can be seen that the Python code finds the parameters fairly accurately and creates the best fit possible to our data. The graphs given in the section before prove that our code succeeds in making the best fitting possible. Especially the Kepler-61b and Kepler-86b light curve fits show this very clearly.

On the other hand, it can also be seen that the results we get from our code match with other publications sufficiently. In order to make a fit to a light curve, this Python code can be used to obtain the best fit possible with the parameters and their errors.

In the future, this code will be worked on. The next goal in this project is to give it a user-friendly interface. Following that, it is intended to be more generalized in terms of working with all the light curves, including the eclipsing binary light curves. Also, more observational data will be used to improve the code and analyze with it.

6. REFERENCES

Fortney, et al., 2011, Discovery and Atmospheric Characterization of Giant Planet Kepler-12B: An Inflated Radius Outlier

Lisa J. Esteves, Ernst J. W. De Mooij & Ray Jayawardhana, 2015, Changing Phases of Alien Worlds: Probing Atmospheres of KEPLER Planets with High – Precision Photometry

Ballard, et al., 2018, Exoplanet Characterization by Proxy: A Transiting 2.15 $R_{\oplus}$ Planet Near the Habitable Zone of the Late K Dwarf Kepler-61

Wang, et al., 2013, Planet Hunters. V. A Confirmed Jupiter-size Planet in the Habitable Zone and 42 Planet Candidates from the Kepler Archive Data

Demircan O., 1990, New Tables of Spherical Eclipse Functions

Claret A., 2000, A new non-linear limb-darkening law for LTE Stellar Atmosphere Models

Demircan O., 2014, Basic Formulae for the Exoplanetary Transit Model Calculation

Mandel K., Agol E, 2002, Analytic Light Curves for Planetary Transit Searches

Selam S., Demircan O., Eliptik Yörüngeli Örten Çift Yıldızların Işık Eğrilerinin Analizi

Special thanks to Prof. Dr. Volkan Bakış, for their guidance and their support.

APPENDIX

A.1-) Python Code for Limb Darkening Constants

In order to obtain the limb darkening constants, a table from Claret (2000) was used. The limb darkening constants were then written on an Excel file. With the Python code, we open the Excel file and start the search for the limb darkening constant for our star.

Like discussed before, every star has its own limb darkening constants that rely on the star's gravity ($\mathbf{\log g}$), effective temperature ($\boldsymbol{T_{\mathrm{eff}}}$), microturbulence ($\boldsymbol{v_{\mathrm{turb}}}$) and metallicity ($[\mathbf{Fe/H}]$). So, in this code, we need these parameters of the star in order to find the constants.

Of course, not every value of $\mathbf{\log g}\,, \boldsymbol{T_{\mathrm{eff}}}, \boldsymbol{v_{\mathrm{turb}}}$ and $[\mathbf{Fe/H}]$ has its own limb darkening constant. For that reason, with the Python code, one can find the closest values possible to the star's parameter values.

For example, let's say $\boldsymbol{T_{\mathrm{eff}} = 5947\ K}$. There isn't a limb darkening constant, especially for this temperature. Because of that, we find the closest value possible to that temperature, which is also included in the limb darkening constants table. In the table, limb darkening constants for $\mathbf{6000}\ \boldsymbol{K}$ exist! As a result, we take the limb darkening constant for $\mathbf{6000}\ \boldsymbol{K}$ which is somewhat close to the $\mathbf{5947}\ \boldsymbol{K}$ temperature anyway.

This needs to be done for every parameter. In the Python code, the algorithm is quite simple; First, find the closest matching parameter values in the table to our real parameter values. Second, search the table for the limb darkening constant that fits all 4 of the parameter values. With this algorithm, we can obtain the limb darkening coefficient for any given star.

A.2-) General formula for $\boldsymbol{C}$ Constants

$$\boxed{C_0 = \frac{1 - \sum_{n=1}^{N} u_n}{1 - \sum_{n=1}^{N} \frac{n u_n}{n+2}}}$$

C_n *constant for* $n = 0$

$$\boxed{C_n = \frac{u_n}{1 - \sum_{n=1}^{N} \frac{n u_n}{n+2}}}$$

C_n *constant for* $n > 0$

A.3-) Calculation of δ Parameter using Orbital Mechanics

$$\boxed{\boldsymbol{M_n = \frac{2\pi}{P}(T_n - T_0) = E_n - e \sin E_n}}$$

Kepler Formula

(Mean Anomaly Formula for Elliptic Orbits)

$\boldsymbol{M_n}$: Mean Anomaly

$\boldsymbol{P}$: Period

$\boldsymbol{T_0}$: Inferior Conjunction

$\boldsymbol{T_n}$: Observation Time

$\boldsymbol{E_n}$: Eccentric Anomaly

$\boldsymbol{e}$: Eccentricity

$$\boxed{\boldsymbol{\tan\frac{\nu_n}{2} = \sqrt{\frac{1+e}{1-e}} * \tan\frac{E_n}{2}}}$$

Relation between the True Anomaly (ν) ile Eccentric Anomaly (E)

$\boldsymbol{E_n}$: Eccentric Anomaly

$\boldsymbol{\nu_n}$: True Anomaly

$\boldsymbol{P}$: Period

$\boldsymbol{e}$: Eccentricity

$$\boxed{\boldsymbol{\delta} = \boldsymbol{\rho}\sqrt{(\sin^2(\boldsymbol{\nu}(\boldsymbol{\theta}) - \boldsymbol{\nu}(\mathbf{0})) * \sin^2 \boldsymbol{i}) + \cos^2 \boldsymbol{i}}}$$

Apparent distance between the exoplanet's center and the star's center for an elliptical orbit

$\boldsymbol{\rho}$: Distance between the components of the system

$\boldsymbol{\theta}$: Phase

$\boldsymbol{\nu}(\boldsymbol{\theta})$: True Anomaly at given Phase

$\boldsymbol{\nu}(\mathbf{0})$: $90 - \omega$

$\boldsymbol{i}$: Inclination

$$\boxed{\boldsymbol{\delta} = \sqrt{(\sin^2 \boldsymbol{\theta} * \sin^2 \boldsymbol{i}) + \cos^2 \boldsymbol{i}}}$$

Apparent distance between the exoplanet's center and the star's center for a circular orbit

$\boldsymbol{\theta}$: Phase

$\boldsymbol{i}$: Inclination

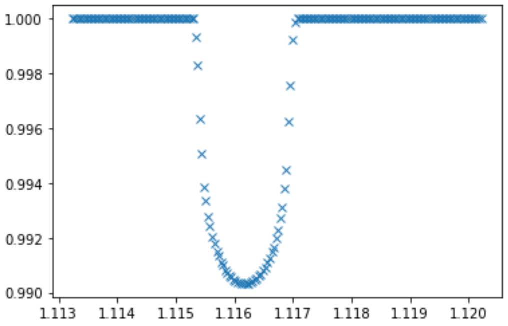
A.4-) Input Parameters and their effect on the Light Curve Fit

The parameters we are dealing with are;

- **Inferior Conjunction (T_0)**
  - Left and Right Shift of the Fit
- **Period (P)**
  - Left and Right Shift of the Fit and the Light Curve!
- **Eccentricity (e)**
  - Left and Right Shift of the Fit (Much larger than T0 Shift)
- **Argument of Periapsis (ω)**
  - Left and Right Shift of the Fit (Also larger than T0 Shift)
- **Inclination (i)**
  - Up and Down Shift of the Fit (Larger than ua and ub Shift)
- r_{planet}
  - Up and Down Shift of the Fit (If it has a lower value, the fit will shift downwards. In other case it will shift upwards)
- r_{star}
  - Changes the Width of the Fit (If it has a higher value, the fit will be wider. In other case it will be narrower)
- **u_a and u_b**
  - Up and Down Shift of the Fit

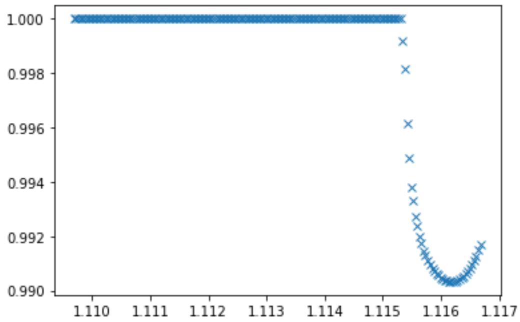
To determine how these parameters affect the fit we are calculating, we will be using Kepler-86b System.

A.4.1-) Inferior Conjunction (T_0)



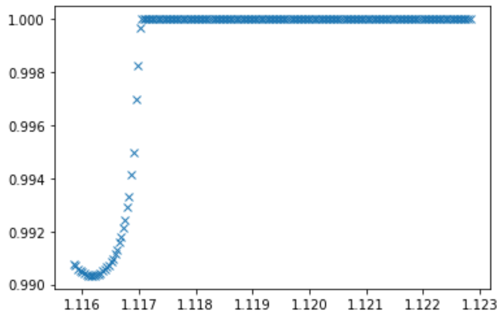
$T_0 = 330.2450$

(Original Value)



$T_0 = 331.2450$

(Higher values of T0 shifts the fit to right)

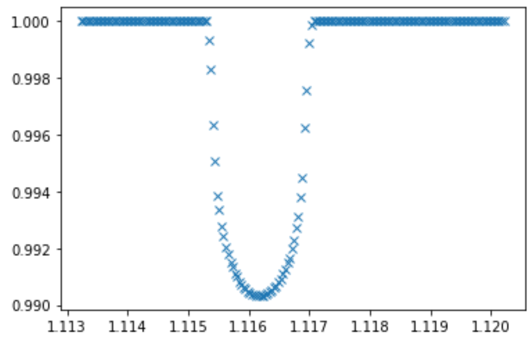


$T_0 = 329.5050$

(Lower values of T0 shifts the fit to left)

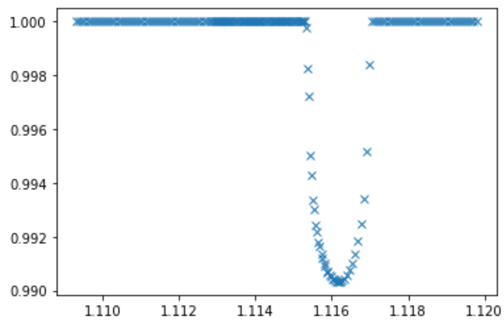
(It can be seen that just by small changes in T0 our fit shifts to left or right drastically but doesn't actually change its position.)

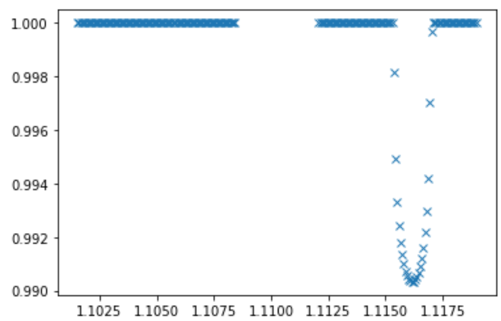
A.4.2-) Period (P)



$$P = 282.5255\ days$$

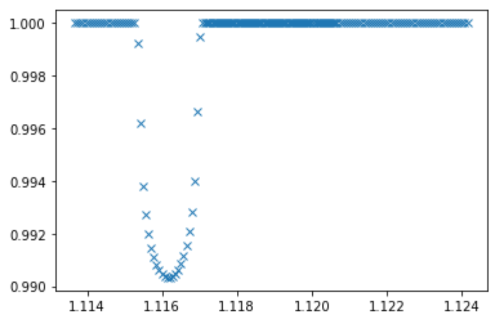
(Original Value)

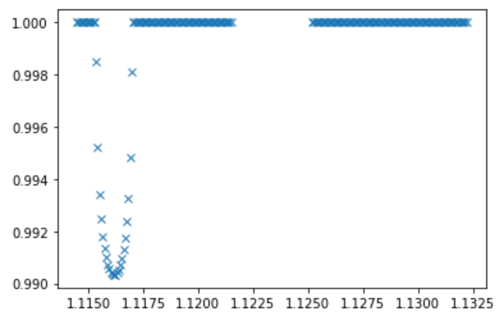




$P = 283.5255\ days$ and $P = 285.5255\ days$

(Higher values of P makes the graph narrower)



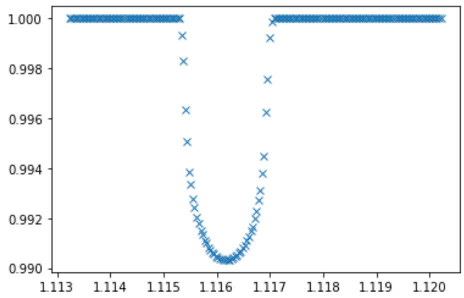


$P = 281.5255\ days$ and $P = 279.5255\ days$

(Lower values of P makes the graph narrower)

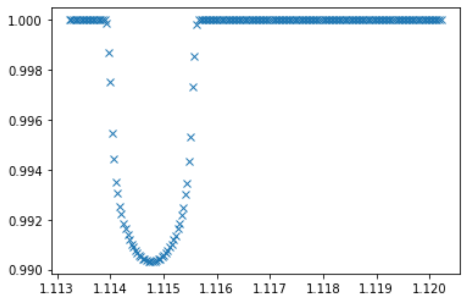
(It can be seen that with changes to P our fit becomes narrower but doesn't actually change its position.)

A.4.3-) Eccentricity (e)



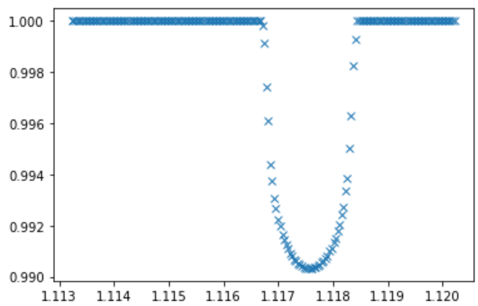
$e = 0.41$

(Original Value)



$e = 0.415$

(Higher values of e shifts the graph to left)

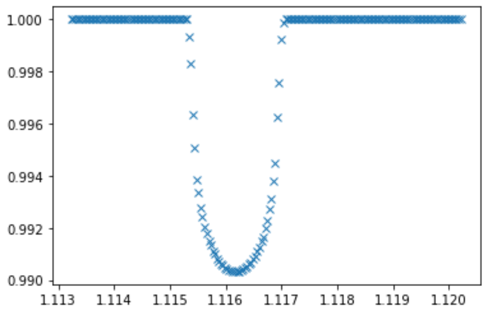


$e = 0.405$

(Lower values of e shifts the graph to right)

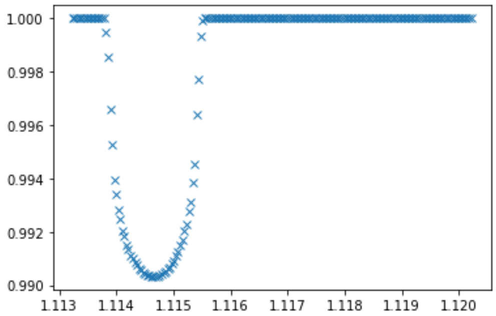
(It can be seen that just by smallest changes in e our fit shifts to left or right much more drastically than T0! Also, this time it does change its position!)

A.4.4-) Argument of Periapsis (ω)



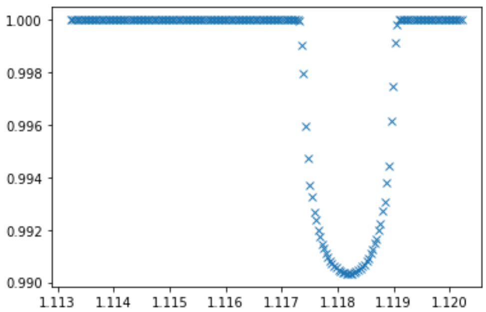
$$\omega = 3°.44$$

(Original Value)



$$\omega = 4°.20$$

(Higher values of ω shifts the graph to left)

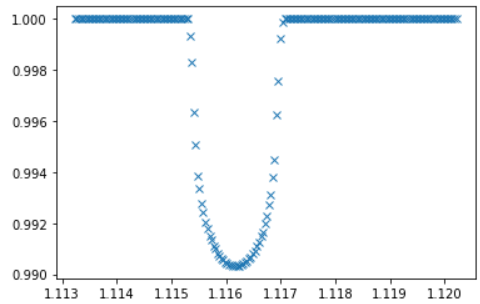


$$\omega = 2°.44$$

(Lower values of ω shifts the graph to right)

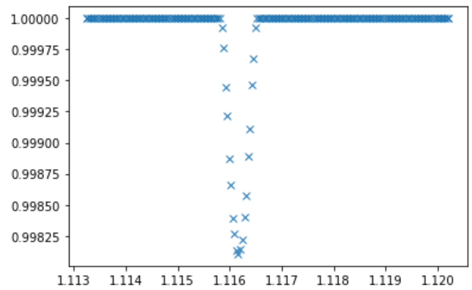
(It can be seen that with changes to ω our fit shifts to left or right)

A.4.5-) Inclination (i)



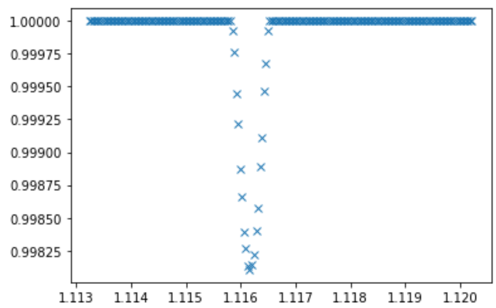
$i = 90°.04$

(Original Value)



$i = 90°.40$

(Higher values of i makes the graph narrower and shallower)

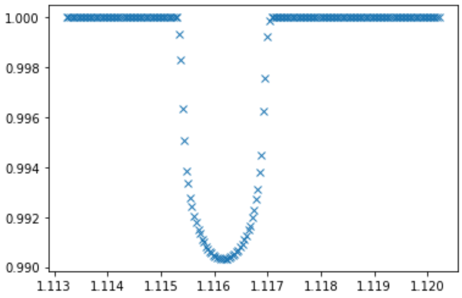


$i = 89°.60$

(Lower values of i makes the graph narrower and shallower)

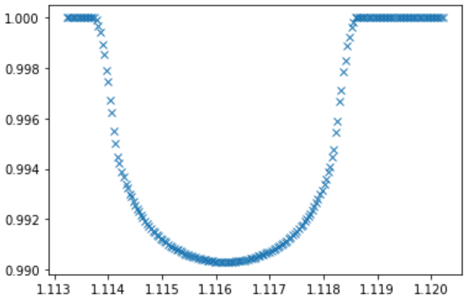
(It can be seen that there is a i value that fits well and the further the value is from the best fitting i value the narrower and shallower the graph will be!)

A.4.6-) Fractional Radius of the Star (r_{star})



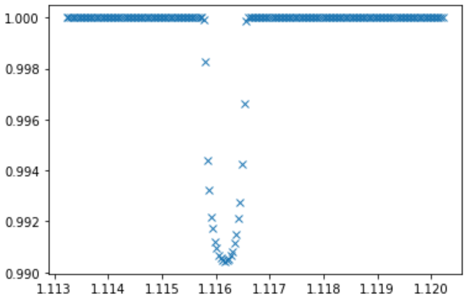
$r_{star} = 0.005659$

(Original Value)



$r_{star} = 0.015659$

(Higher values of r_{star} makes the graph wider)

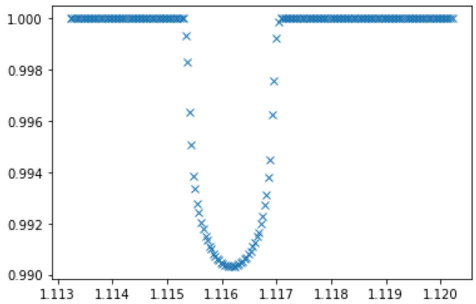


$r_{star} = 0.002659$

(Lower values of r_{star} makes the graph narrower)

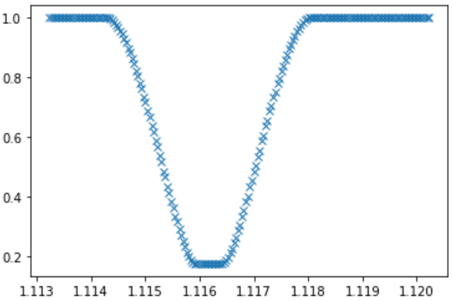
(It can be seen that just by small changes in r_{star} our fit becomes wider or narrower drastically! But deepness does not change.)

A.4.7-) Fractional Radius of the Planet (r_{planet})



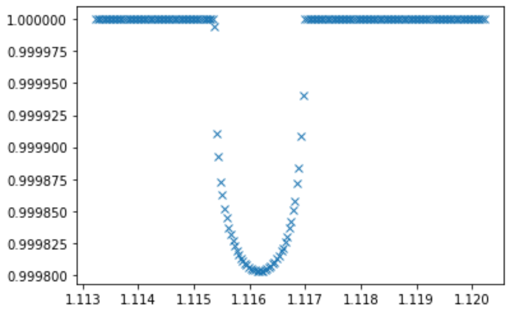
$r_{planet} = 0.0005365$

(Original Value)



$r_{planet} = 0.0075365$

(Higher values of r_{planet} makes the graph deeper)

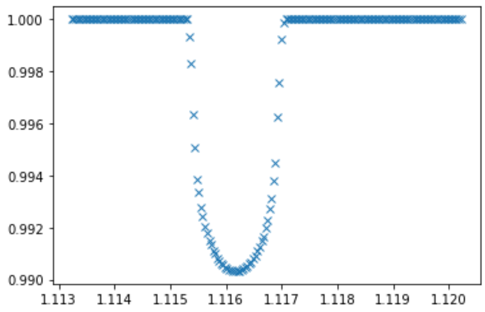


$r_{planet} = 0.0000765$

(Lower values of r_{planet} makes the graph shallower)

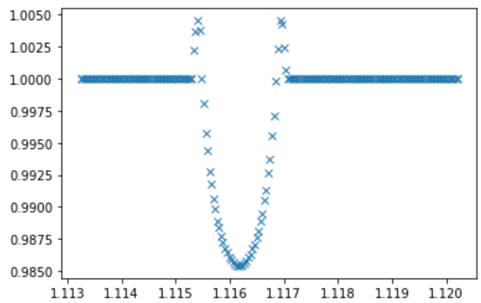
(It can be seen that just by changes in r_{planet} our fit becomes deeper or shallower!)

A.4.8-) Limb Darkening Coefficients (u_a and u_b)



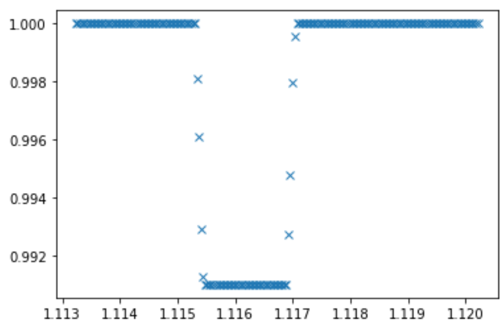
$u_a = 0.31$ and $u_b = 0.30$

(Original Value)



$u_a = 0.60$ and $u_b = 0.60$

(Higher values of u_a and u_b makes the graph lose its true shape)



$u_a = 0$ and $u_b = 0$

(Lower values of u_a and u_b makes the graph more linear)

(It can be seen that just by changes in u_a and u_b our fit's shape changes!)