Finding exoplanets using python and TESS data

R. Virinchi I20PH008
NIT Surat

May 2023

1 Introduction

1.1 lightkurve

Lightkurve is an open-source Python package designed for working with time-series data from space telescopes, particularly in the field of astronomy. It provides tools and functionalities for the analysis of light curves, which are plots of brightness over time, often used in the study of variable stars, exoplanets, and other astronomical phenomena. Lightkurve is especially well-suited for data from missions like NASA's Kepler, TESS, and other similar space telescopes.

1.2 TESS

The Transiting Exoplanet Survey Satellite (TESS) is a NASA space telescope mission designed to discover exoplanets orbiting the brightest stars in the sky. Employing the transit method, TESS systematically observes large sectors of the sky for extended periods, dividing it into 26 sectors and monitoring each for about 27 days. Equipped with four wide-field cameras known as the TESS Cameras, the satellite captures temporary brightness dips in stars as planets pass in front of them. TESS follows a highly elliptical orbit facilitating efficient data transfer and continuous monitoring. Data is made publicly accessible through the Mikulski Archive for Space Telescopes (MAST). Beyond exoplanet discoveries, TESS contributes to stellar astrophysics, providing insights into stellar activity and astrophysical phenomena. Its success has led to the extension of its mission, demonstrating its crucial role in advancing our understanding of planetary systems beyond our solar system.

2 Python code

2.1 Importing modules

```
from lightkurve import search_targetpixelfile
from lightkurve import TessTargetPixelFile
```

```
import lightkurve as lk
import numpy as np
```

2.2 Downloading files

```
pixelFile = search_targetpixelfile('KIC 6922244',
author="Kepler",
cadence="long", quarter=4).download()
# Show a single snapshot
pixelFile.plot(frame=42)

from urllib.request import urlopen
from io import BytesIO
from zipfile import ZipFile

def download_and_unzip(url, extract_to='.'):
    http_response = urlopen(url)
    zipfile = ZipFile(BytesIO(http_response.read()))
    zipfile.extractall(path=extract_to)

product_group_id = '27240036'
url =
'https://mast.stsci.edu/api/v0.1/Download/bundle.zip?
previews=false&obsid=' + product_group_id
destination = '/TESS/'

download_and_unzip(url, destination)
```

2.3 Finding patter and period of rotation

```
import matplotlib.pyplot as plt
%matplotlib widget

# We'll combine the individual frames into a lightcurve
# Aperture masks make the image look better for analysis
lc =
pixelFile.to_lightcurve(aperture_mask=pixelFile.pipeline
_mask)
lc.plot()

# We may find it easier to spot the pattern if we
flatten the curve
flat_lc = lc.flatten()
flat_lc.plot()

# Phase-fold the light curve to verify that the period
```

```
and transit time
# correspond to the transit signal
# This puts the frequency spikes on top of each other
if we get the period right
folded_lc = flat_lc.fold(period=3.5225)
folded_lc.plot()

# we usse a periodogram to show all the repetitive patterns
in our graph
# Gives us the most likely candidate

# Periodograms are an estimate of the spectral density
of a signal
# (An estimation of what the Fourier Transform of the
data would look
like
#  if it were a continuous function

period = np.linspace(1, 5, 10000)
# BLS = Box Least Squares
bls = lc.to_periodogram(method='bls', period=period,
frequency_factor=500)
bls.plot()

# Period value corresponding to the highest peak in the
periodogram
planet_x_period = bls.period_at_max_power

planet_x_t0 = bls.transit_time_at_max_power
planet_x_dur = bls.duration_at_max_power

# Folding can yield a lot of information about the
planet
# The depth can tell us about the size, etc
ax = lc.fold(period=planet_x_period,
epoch_time=planet_x_t0).scatter() ax.set_xlim(-2,2)

print(planet_x_period)
print(planet_x_t0)
print(planet_x_dur)

plt.show()
```

2.4 Output

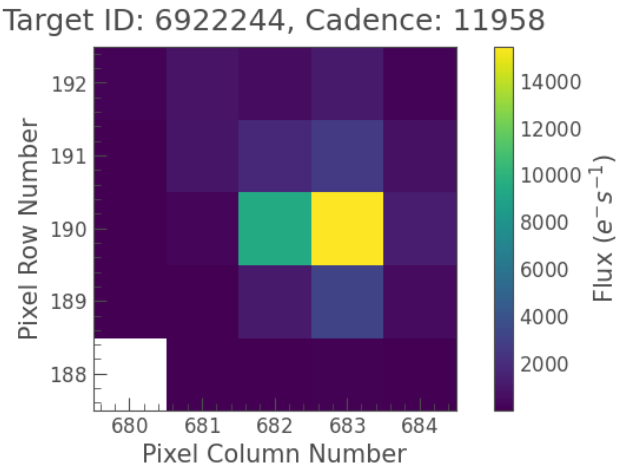

Figure 1: Pixel image

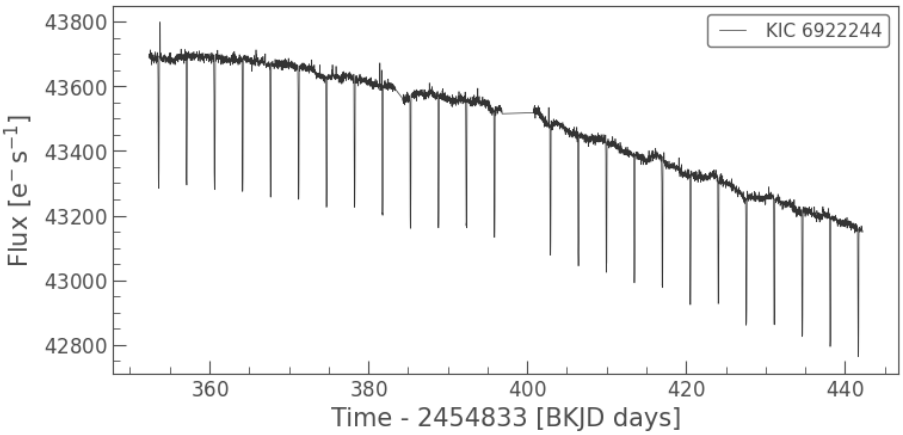

Figure 2: Time vs Flux

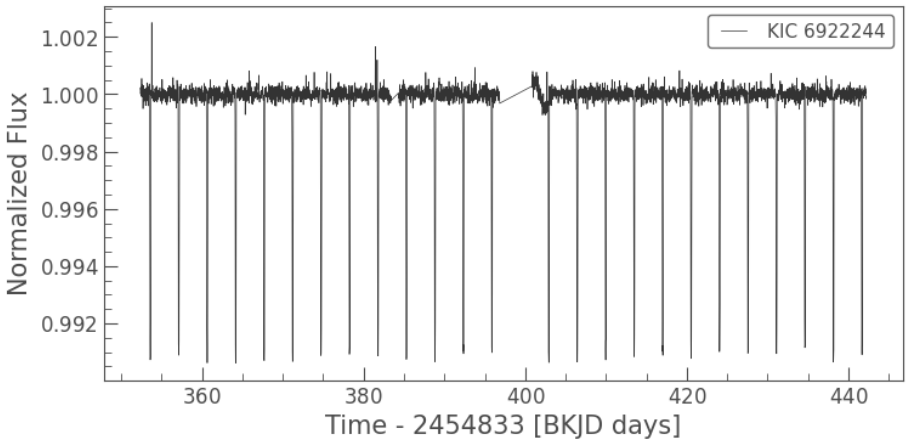

Figure 3: Time vs Flux flattened

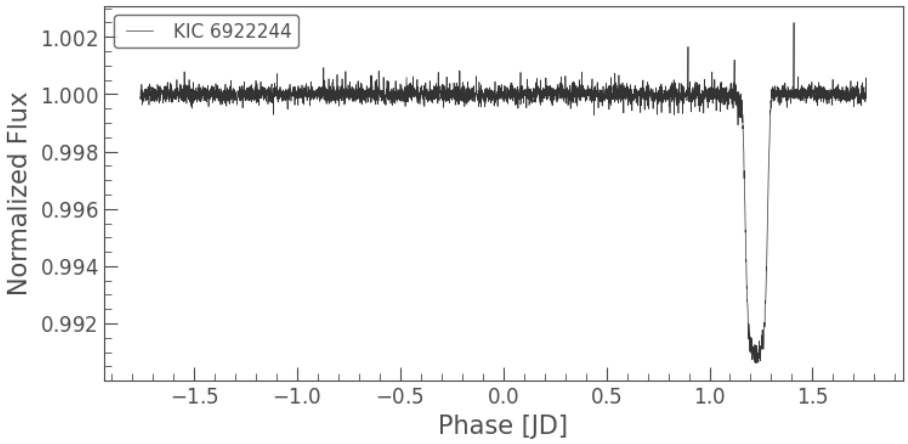

Figure 4: Phase vs Normalized flux

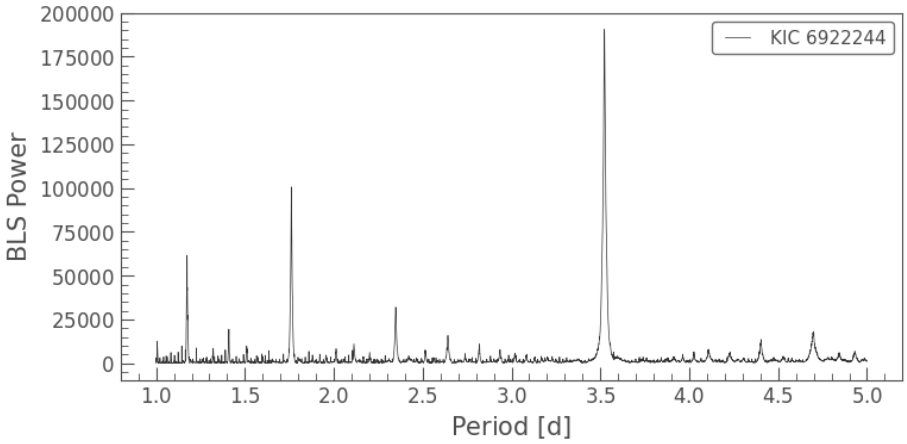

Figure 5: Period vs BLS power

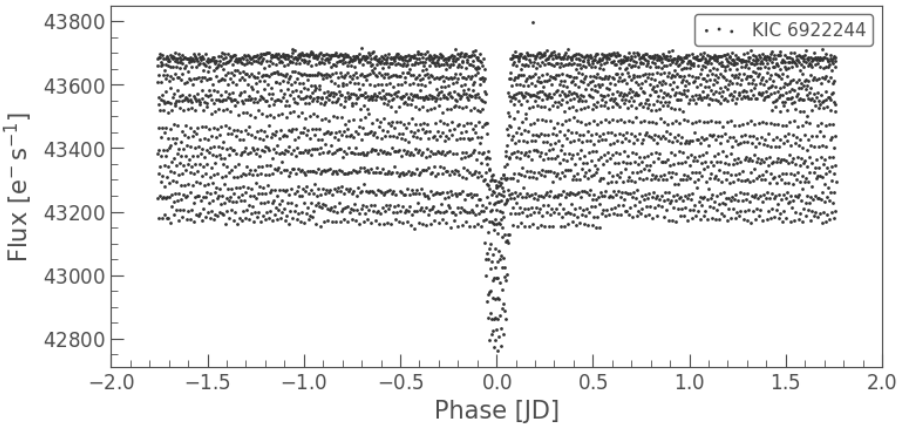

Figure 6: Phase vs Flux

3 Summary

To find exoplanets using the Lightkurve package in Python, start by installing the package with the command `pip install lightkurve`. Once installed, import the necessary modules in your script or Jupyter Notebook using `import lightkurve as lk`. Next, download and load time-series data from a space telescope, such as Kepler or TESS, using the `lk.search_targetpixelfile` and `download` functions. Normalize the light curve with `tpf.to_lightcurve() .normalize()`. To enhance the clarity of transits, detrend the light curve by flattening it using `lc.flatten(window_length=401).remove_outliers()`. Following this, employ the periodogram method with the Box Least Squares (BLS) algorithm to identify the most likely period of the transiting exoplanet. Access the best period using `periodogram.period_at_max_power`. Proceed to fold the light curve based on the identified period to visualize the transits more effectively. Use `lc.fold(period = best_period, t0 = periodogram.transit _time_at_max_power)` for this purpose. Identify the transits within the folded light curve using `folded_lc .extract_transit_times()`. For a comprehensive analysis, plot the original light curve with `lc.plot()`, and visualize the folded light curve with identified transits using `folded_lc .scatter()`. Adjust parameters and methods as needed for your specific data and target.

4 Conclusion

From the above graphs we can conclude that there is an exoplanet orbiting the star we studied.