Practice I : Manipulating TESS Data

Ana Pereira Aguirre

May 11, 2025

---

1 Introduction

In the era of modern astronomy, large-scale surveys generate unprecedented volumes of data on a daily basis. For instance, the Atacama Large Millimeter/submillimeter Array (ALMA) is generating 1 terabyte of scientific data per day[1], while space-based missions like the Transiting Exoplanet Survey Satellite (TESS) have yielded hundreds of terabytes of photometric observations, along with thousands of identified TESS Object of Interest (TOIs)[2]. As these data repositories grow, accessing and processing individual targets of interest manually through web interfaces becomes increasingly inefficient. It is therefore necessary to resort to command-line tools-such as those available in Unix-based environments-to automate the querying, downloading, and visualization of time-series data. In this context, the process leading up to the plotting of TESS light curves offers a valuable opportunity to understand and apply automation tools for handling large-scale astronomical data, particularly within Unix-based environments.
The Transiting Exoplanet Survey Satellite (TESS) is an all-sky transit survey whose main goal is to detect Earth-sized planet masses and atmospheric compositions[3]. TESS collects time-series photometry at various cadences, with durations ranging from $\sim$27 days to a year depending on the overlap of observational sectors. Full Frame Images (`FFIs`), which capture the entire CCD area, are collected every 30 minutes. Additionally, for a subset of selected targets, smaller pixel regions centered on the stars are extracted at a higher cadence-every 2 minutes and 20 seconds-through Target Pixel Files (`TPFs`), which store image stamps around each star per step[4].

Each TPF contains a time-series of small cutouts, enabling the monitoring of a star's brightness over time. By applying aperture photometry to these sequences of images, the integrated flux from each frame is measured, generating a time-series of flux values. This process produces Light Curve (`LC`) files, which include multiple data outputs, such as simple aperture photometry (`SAP_FLUX`), a systematics-corrected version (`PDCSAP_FLUX`), position vectors, and quality flags. The `PDCSAP_FLUX` light curve is particularly useful for the detection of transits, as it removes common instrumental trends and corrects for flux losses due to crowding and aperture effects. The full process-ranging from the acquisition of FFIs to the generation of calibrated light curves-is illustrated in the figure 1.

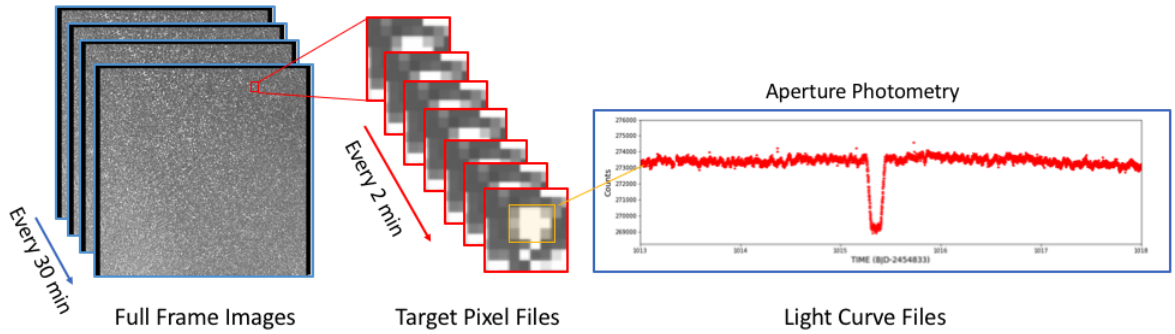


Figure 1: Schematic of the data flux in TESS: from Full Frame Images, through the Target Pixel Files, to the light curves generated by aperture photometry. Source: MAST/STScI[5].

The following sections describes the methodology used to download, organize and plot TESS light curve files using Unix-based command-line tools.

2 Methodology and Data Acquisition

To being working with TESS light curves, the first step involved downloading the relevant time-series data. This was achieved by using Bash script to automate the retrieval of pre-packaged download scripts hosted on STScI's Mikulski Archive. Specifically, the target was Sector 73 light curves, which are available in FITS format. The process started by downloading the script "`tesscurl_sector_73_lc.sh`", which contains a list of individual curl commands, each pointing to a light curve file. The following Bash script was used to retrieve and execute this download script:

- We create the script using `nano` editor:

```
nano download_tess_lc.sh
```

- Inside the script, we write:

```
#!/bin/bash
# To download files from a link we use
url="https://archive.stsci.edu/missions/tess/
download_scripts/sector/"

# We name what we're looking for
light_curve="tesscurl_sector_73_lc.sh"

# To download, we use

curl -O "${url}/${light_curve}"

# To confirm the download

echo "Downloaded file: ${light_curve}"

# To downloaded the first 20 files

head -n 21 "${light_curve}" | bash

echo "Partial download completed (20 files)"

```

- After saving the changes in our new script, we use the command `chmod 755` to make the script executable:

```
chmod 755 download_tess_lc.sh
```

- We then run the script using:

```
./download_tess_lc.ch
```

As a result, we obtain 20 FITS files, as shown in Figure 2:

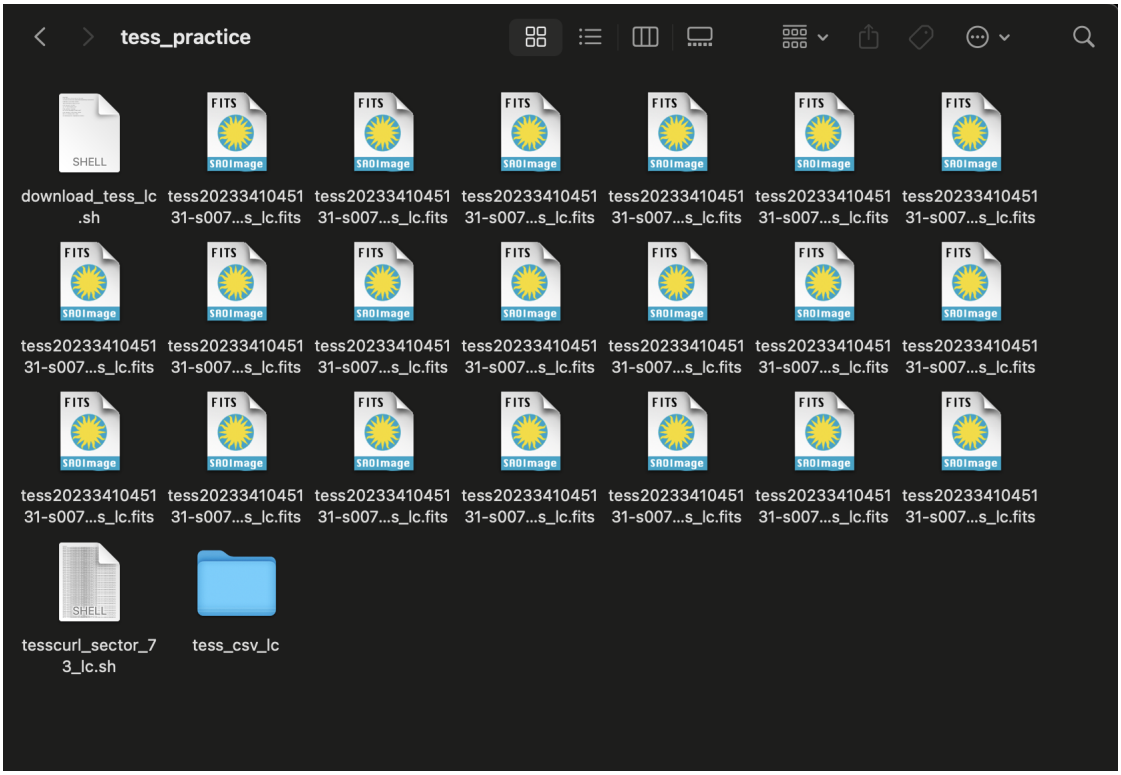


Figure 2: Light Curves FITS files obtained from STScI's Mikulski Archive.

2.1 FITS to CSV Conversion

Once the FITS files were downloaded, the next step involved converting them to CSV format. To streamline the process, all FITS files located in the working directory were opened simultaneously by executing the following command in the terminal:

```
topcat /Users/anapereira/Documents/tess_practice/*.fits
```

Once loaded into TOPCAT, each table was manually exported to CSV format through the "Save Table(s)" option available in the session menu. Each output file was renamed using the last digits of the original FITS filename for consistency and ease of reference. Both the file path and naming convention are illustrated in Figure 3.

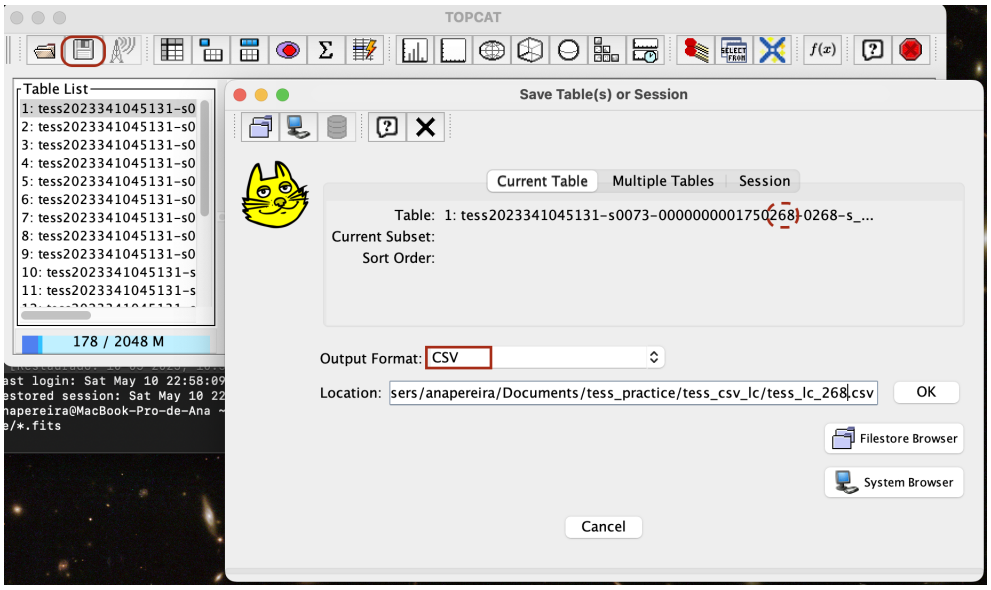


Figure 3: FITS file exported to CSV using TOPCAT.

2.2 Organizing File Names

One of the key advantages of using a Unix-based shell is the ability to efficiently manipulate files and directories through scripted commands. This includes operations such as listing, copying, renaming, and moving files-tasks that would otherwise be working with large astronomical datasets like those produced by TESS. In this part of the assignment, this capability was leveraged to automate the processing and organization of light curve files.

To begin, a script named `find_tess.sh` was created and made executable (following the method detailed at the beginning of Section 2). The purpose of this script was to generate a text file containing the names of all CSV files within the working directory.

- The directory is located at:

```
/Users/anapereira/Documents/tess_practice/tess_csv_lc/
```

- Inside the script, a variable was defined to hold the name of the output file:

```
tess_names="tess_csv.txt"
```

- The following command lists all CSV files and stores their names in the variable:

```
ls *.csv > "$tess_names"
```

- To verify the results, the content of the output file was printed:

```
echo "Files found $tess_names:"
cat $tess_names
```

- Executing the script produced the list shown in Figure 4.

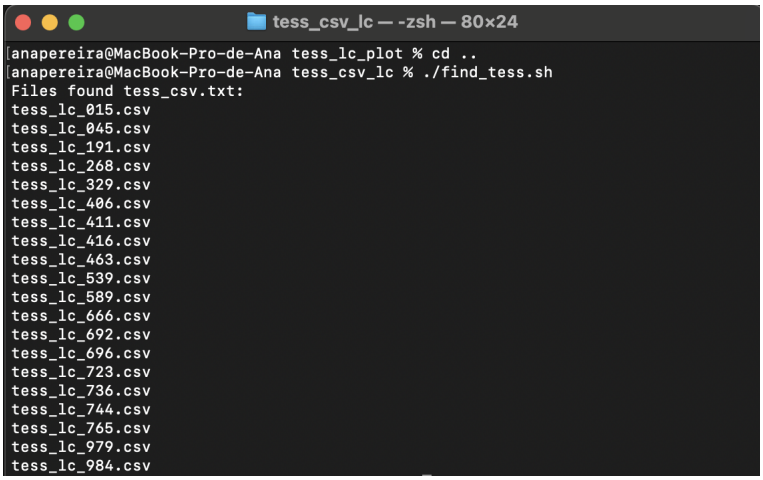


Figure 4: Output displaying the list of CSV file names found by the script.

To complete the next task, which involves splitting the list into files containing only five names each:

- Another executable script was created, named `tess_five.sh`. This script first initialized new filenames for the subdivision:

```
archivo1="tess_part1.txt"
archivo2="tess_part2.txt"
archivo3="tess_part3.txt"
archivo4="tess_part4.txt"
tess="tess_csv.txt"
```

- Then, using a combination of `head` and `tail`, the original list was divided into four parts, each containing five file names:

```
head -n 5 "$tess" | head -n 5 > "$archivo1"
tail -n +6 "$tess" | head -n 5 > "$archivo2"
tail -n +11 "$tess" | head -n 5 > "$archivo3"
tail -n +16 "$tess" | head -n 5 > "$archivo4"
```

- Finally, the content of each subdivision was printed to confirm successful execution:

```
echo "First 5 CSV file names"
cat "$archivo1"
echo ""
echo "Second csv files names division"
cat "$archivo2"
echo ""
echo "Third csv files names division"
cat "$archivo3"
echo ""
echo "Fourth csv files names division"
cat "$archivo4"
```

- The resulting terminal output is illustrated in Figure5.

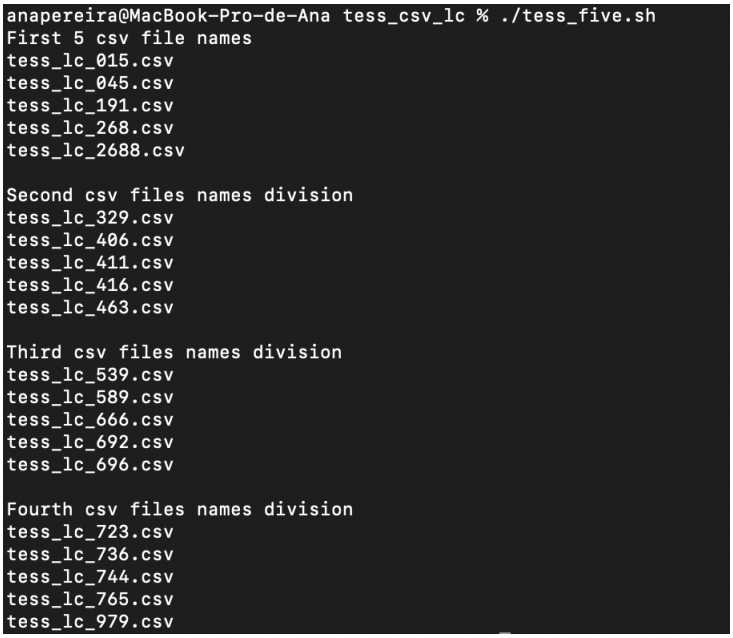


Figure 5: Output displaying the list of CSV file names finding by the script.

Although there are simpler ways to accomplish this task, such as using the `split` command, we chose to take this more detailed approach based on what we learned in the tutorial session.

3 Light Curve Visualization

To visualize the TESS light curves, the CSV files previously generated were opened in `TOPCAT`. This was accomplished by navigating to the working directory `tess_csv_lc` and executing the following command in the terminal:

```
topcat *.csv
```

Once the files were loaded, the PLANE PLOT button in TOPCAT was used to generate the light curves. For each file, the x-axis was set to the time column, expressed in BTJD (Barycentric TESS Julian Date), and the y-axis was set to the normalized flux column PDCSAP_FLUX, which is measured in electrons per second (e/s), as shown in the figure 6.

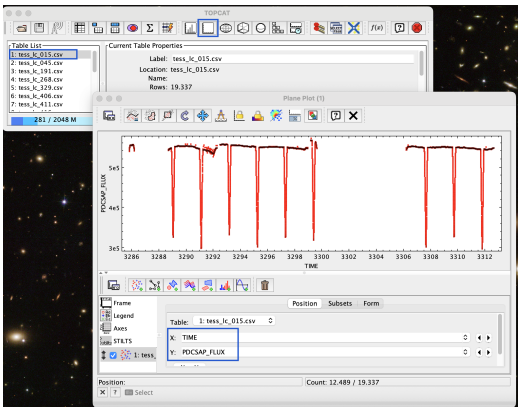


Figure 6: Example of a light curve plotted in TOPCAT using the time column (BTJD) and the normalized flux (PDCSAP_FLUX). The plot was exported as a high-resolution PDF.

The choice of PDCSAP_FLUX over SAP_FLUX is motivated by its improved quality for exoplanet detection. As explained in the introduction, SAP_FLUX is derived by summing the pixel fluxes within a predefined aperture, which may still contain systematic artifacts. In contrast, PDCSAP_FLUX is processed through the Presearch Data Conditioning (PDC) pipeline, which corrects for instrumental systematics while preserving astrophysical signals such as transits[6].
After configuring the axes and titles for each light curve, the plots were saved individually using the "Export Plot" option in TOPCAT, selecting the PDF format to ensure high—resolution output. This procedure is illustrated in Figure 7.

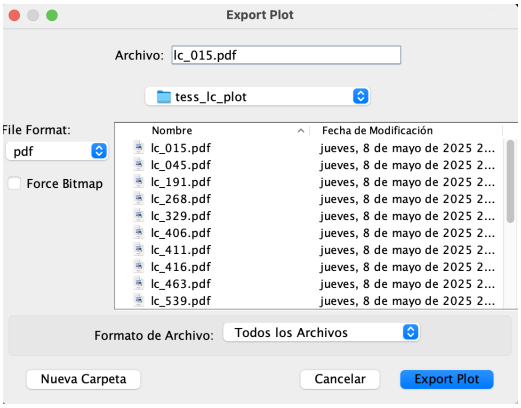
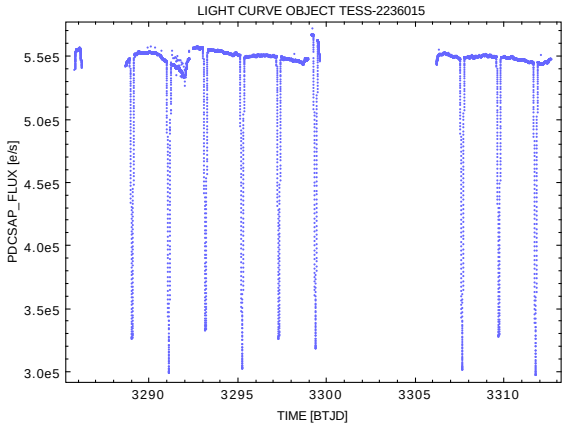


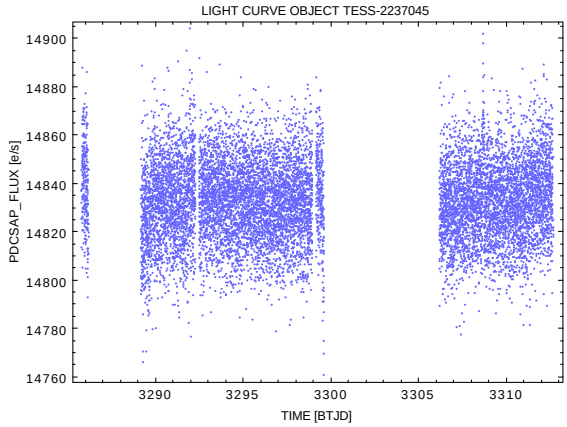
Figure 7: Interface for exporting light curve plot in TOPCAT. Each plot is saved as a PDF file, allowing for consistent formatting and high-resolution output of the analyzed data.

4 Results

After exporting all the light curves from `TOPCAT` in high-resolution `PDF` format, a selection of four examples was chosen to be presented in this section. The remaining plots are in Appendix A for reference and completeness.

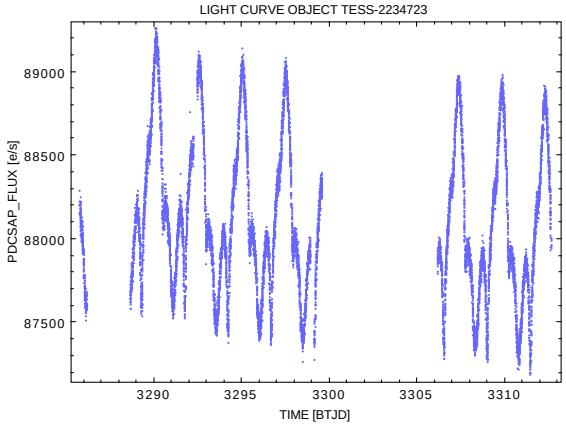


((a)) Light curve of TESS object 015.

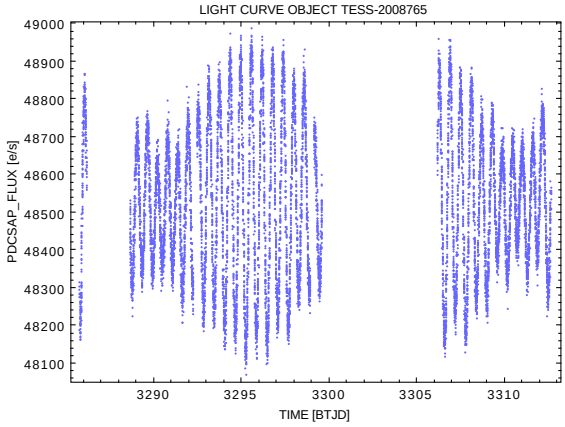


((b)) Light curve of TESS object 045.

Figure 8: The flux variability is plotted as a function of the time `BTJD`), with data extracted from `PDSAP_FLUX` values.



((a)) Light curve of TESS object 723.



((b)) Light curve of TESS object 765.

Figure 9: The flux variability is plotted as a function of the time `BTJD`), with data extracted from `PDSAP_FLUX` values.

5 Discussion and Conclusion

Today, handling large volumes of data is an essential skill for any astronomical research, especially with the technological advancements that allow us to obtain massive data from space. One of the main conclusions of this practice exercise is that optimizing techniques for processing and analyzing this data is crucial. In that regard, although tools like `TOPCAT` are useful, the efficiency of the process can be significantly improved by using programming languages like `Python`.

In the case of `TOPCAT`, one of the main drawbacks is that it requires manual manipulation of files. For example, each FITS file must be converted to CSV before it can be processed, which becomes a tedious task when dealing with large datasets. Additionally, creating and saving the plots must be done manually for each individual light curve, which represents a considerable amount of time and effort.
In contrast, with Python, these processes can be automated, significantly reducing the time spent on repetitive tasks. Python allows for more efficient file manipulation and offers powerful libraries like `Matplotlib` for data visualization. By automating the conversion of files, creation of plots, and extraction of relevant features from the light curves, the time invested is significantly reduced. However, it is important to acknowledge that `TOPCAT` has an advantage in terms of accessibility. The main appeal of `TOPCAT` lies in its graphical interface, which allows for file manipulation without the need to write code. This makes it a suitable tool for users who prefer to work interactively and visually.
Within the light curves, an interesting observation is that, although they come from the same sector, we can find a significant variety of photometric behaviors. Each light curve carries a different meaning, and the ability to distinguish between these types is crucial for identifying astronomical phenomena such as planetary transits, stellar variability, or noise.

Finally, with the increasing amount of data being received daily from space missions, it is clear that more efficient and automated techniques are needed to process and analyze this vast amount of information. As astronomy continues to advance, the need to optimize workflows and reduce processing time becomes increasingly crucial to fully harness the data being collected.

References

[1] Fabrizia Guglielmetti et al. *Bayesian and Machine Learning Methods in the Big Data era for astronomical imaging*. arXiv:2210.01444 (2022). https://arxiv.org/abs/2210.01444

[2] Wikipedia contributors.

Transiting Exoplanet Survey Satellite. Wikipedia, The Free Encyclopedia. Available at: https://en.wikipedia.org/wiki/Transiting_Exoplanet_Survey_Satellite. Accessed: May 11, 2025.

[3] Mikulski Archive for Space Telescopes (MAST). *TESS Mission Archive*. Available at: https://archive.stsci.edu/missions-and-data/tess. Accessed: May 11, 2025.

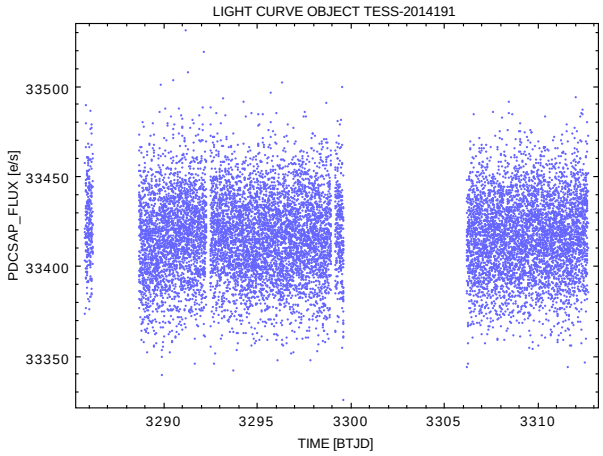
[4] TESS Science Support Center. *TESS Data Products Overview, Version 2.0*. STScI Outerspace. Available at: https://outerspace.stsci.edu/display/TESS/2.0+-+Data+Product+Overview. Accessed: May 11, 2025.

[5] TESS Science Support Center. "TESS Data Products Overview", STScI. https://outerspace.stsci.edu/display/TESS/2.0+-+Data+Product+Overview. Accessed May 2025.

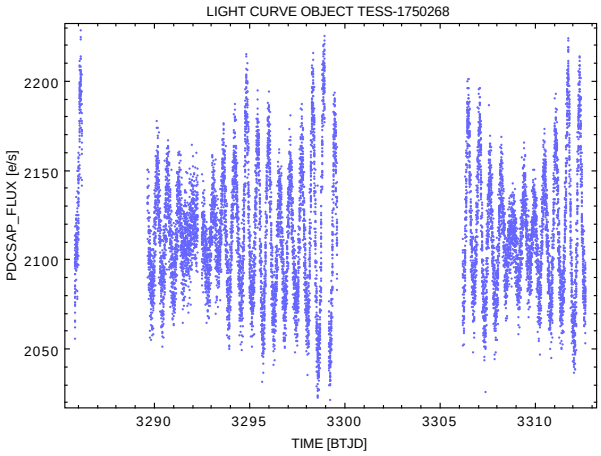
[6] NASA HEASARC. *TESS Light Curve File Object Tutorial*. Available at: https://heasarc.gsfc.nasa.gov/docs/tess/LightCurveFile-Object-Tutorial.html. Accessed: May 11, 2025.

A Additional Light Curves

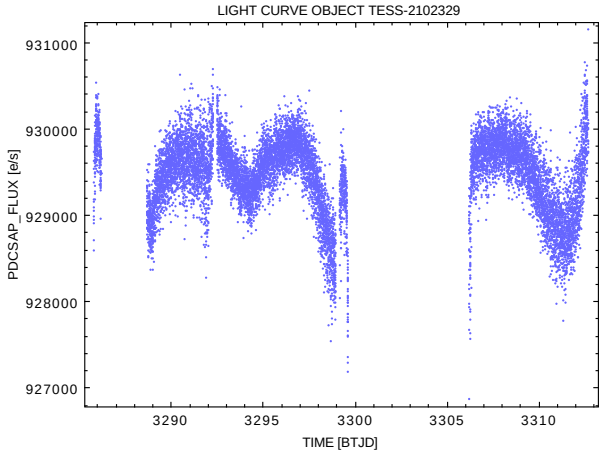
The following figures present the additional light curves generated from TESS data in Sector 73, as part of the data processing and visualization activity. Each plot display the flux variation of a selected object over time, using `PDCSAP_FLUX` values.



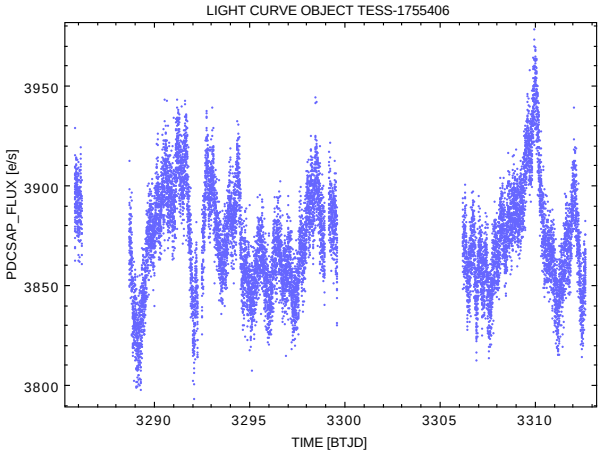
((a)) Light curve of TESS object 191.



((b)) Light curve of TESS object 268.

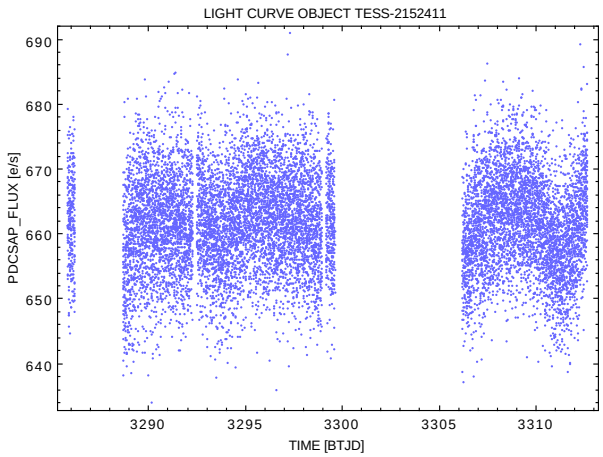


((c)) Light curve of TESS object 329.

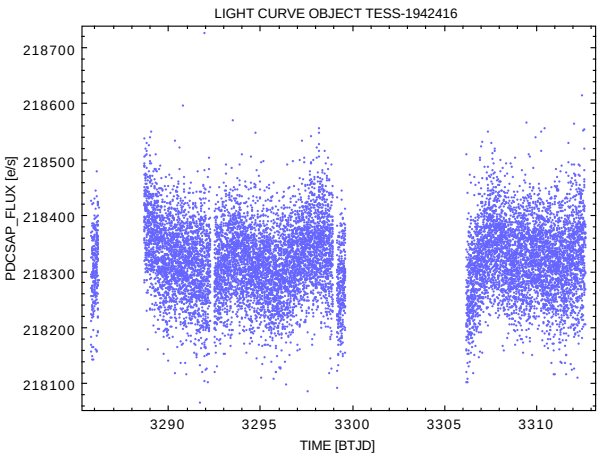


((d)) Light curve of TESS object 406.

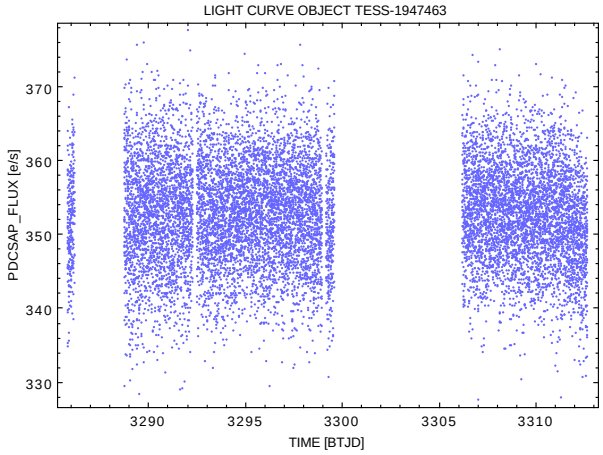
Figure 10: Light curves of selected TESS objects. Each plot shows the flux variability over time, with the x-axis representating `BTJD` and y-axis respresentating the normalized `PDCSAP_FLUX`.



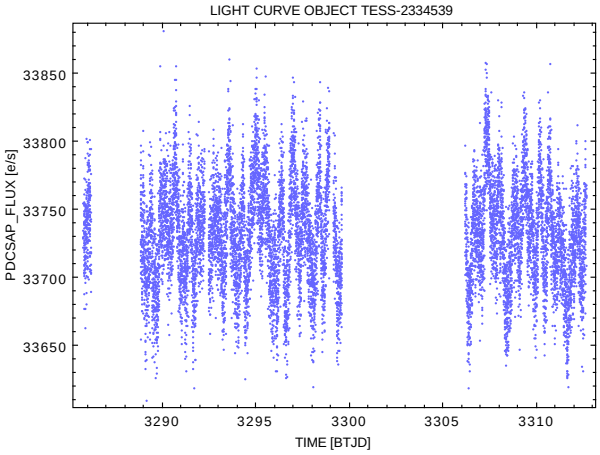
((a)) Light curve of TESS object 411.



((b)) Light curve of TESS object 416.

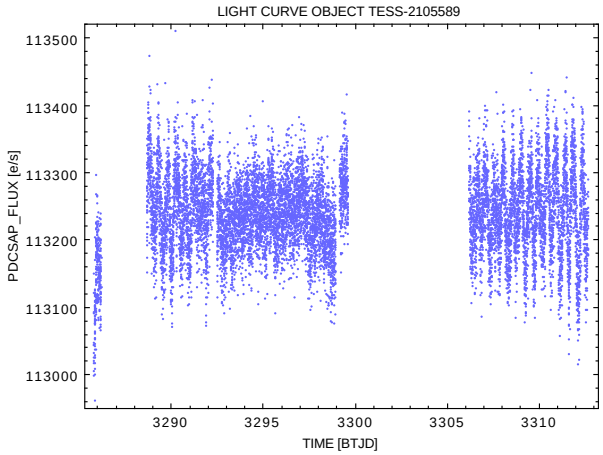


((c)) Light curve of TESS object 463.

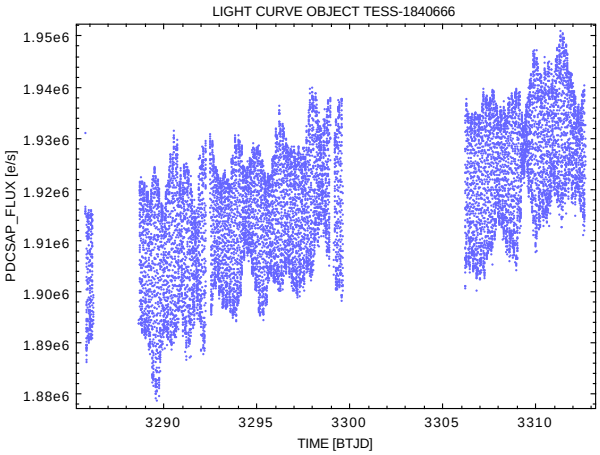


((d)) Light curve of TESS object 539.

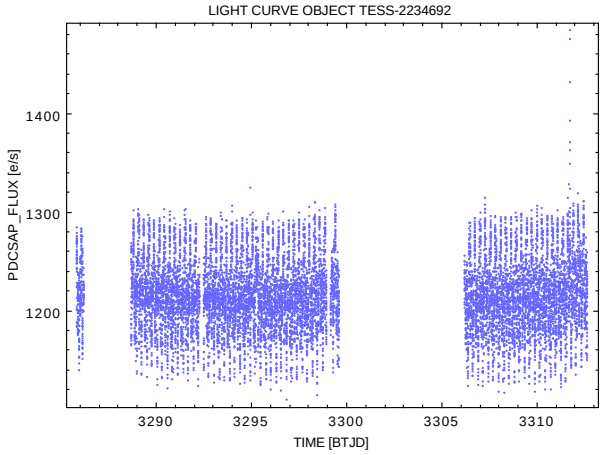
Figure 11: Light curves of selected TESS objects. Each plot shows the flux variability over time, with the x-axis representating `BTJD` and y-axis respresentating the normalized `PDCSAP_FLUX`.



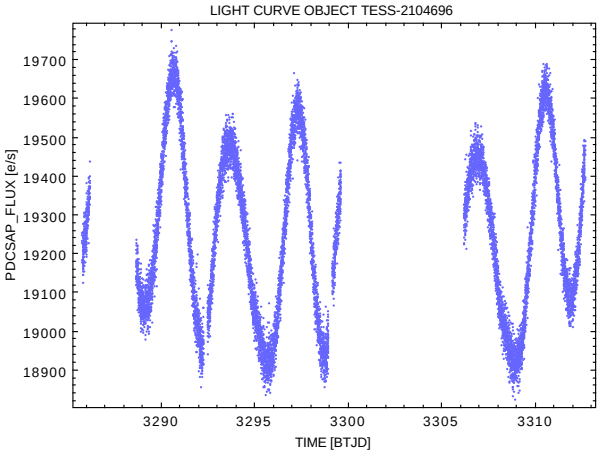
((a)) Light curve of TESS object 589.



((b)) Light curve of TESS object 666.

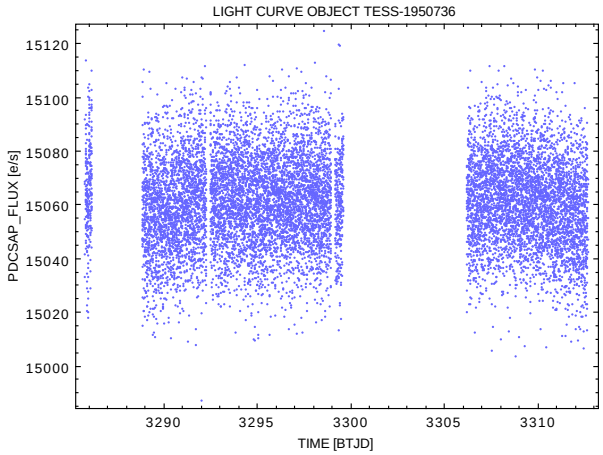


((c)) Light curve of TESS object 692.

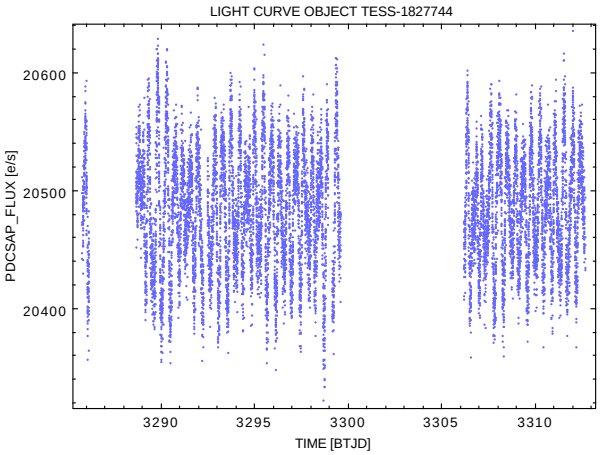


((d)) Light curve of TESS object 696.

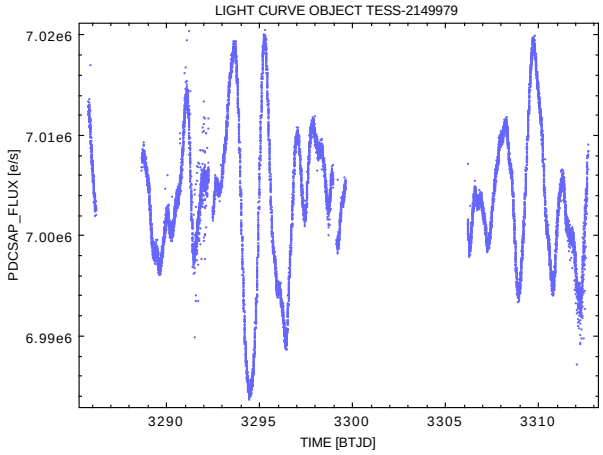
Figure 12: Light curves of selected TESS objects. Each plot shows the flux variability over time, with the x-axis representating `BTJD` and y-axis respresentating the normalized `PDCSAP_FLUX`.



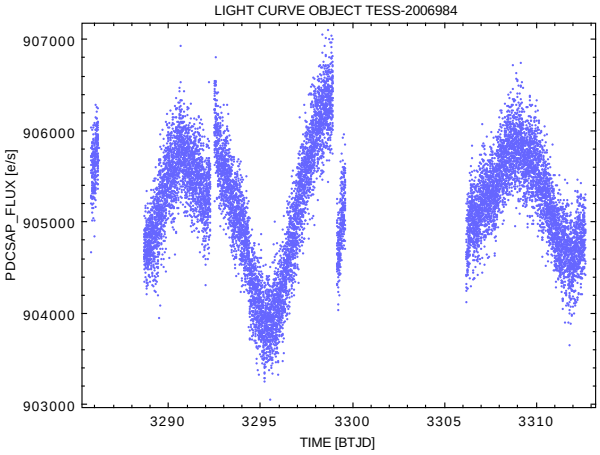
((a)) Light curve of TESS object 736.



((b)) Light curve of TESS object 744.



((c)) Light curve of TESS object 979.



((d)) Light curve of TESS object 984.

Figure 13: Light curves of selected TESS objects. Each plot shows the flux variability over time, with the x-axis representating `BTJD` and y-axis respresentating the normalized `PDCSAP_FLUX`.