DEVELOPING ASTROPHYSICAL DATA MANAGEMENT & PROCESSING TOOLS 1

STUDENTS / UNIVERSITIES

İlker GÜL
Melike Sena ÖZGEN
Mehmet Akif GÜLTEKİN

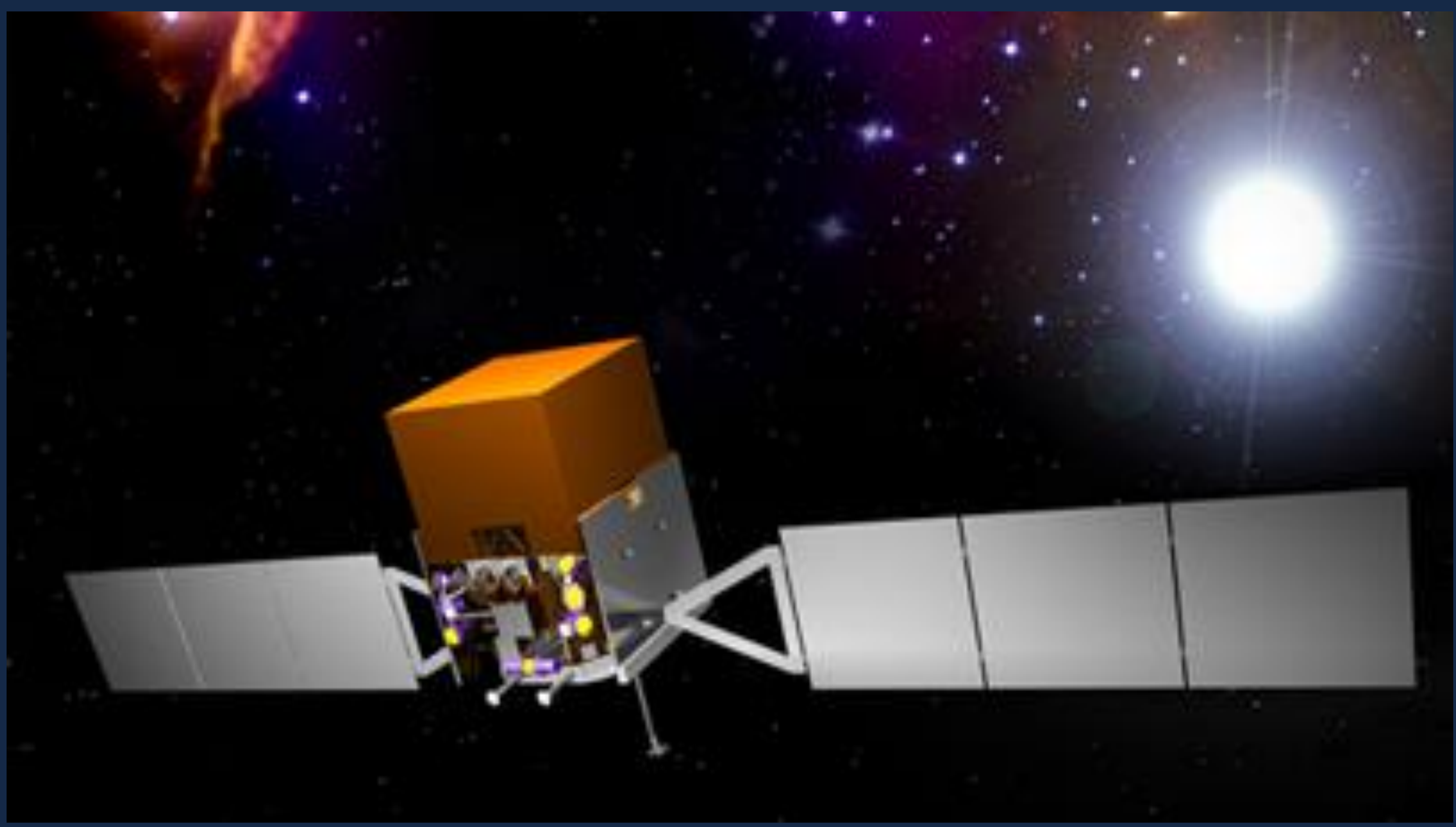
SUPERVISOR(S)

Yuki Kaneko GÖĞÜŞ
Ersin GÖĞÜŞ
Can GÜNGÖR

Sabancı Üniversitesi

PURE – Program for Undergraduate Research

INTRODUCTION



For more than 10 years; Supermassive black holes, merging neutron stars, streams of hot gas moving close to the speed of light and other phenomena in which *gamma-ray radiation* is generated, which is the most energetic form of electromagnetic radiation, can be observed by Fermi Gamma-ray Space Telescope ("Fermi Gamma-ray Space Telescope," 2019).

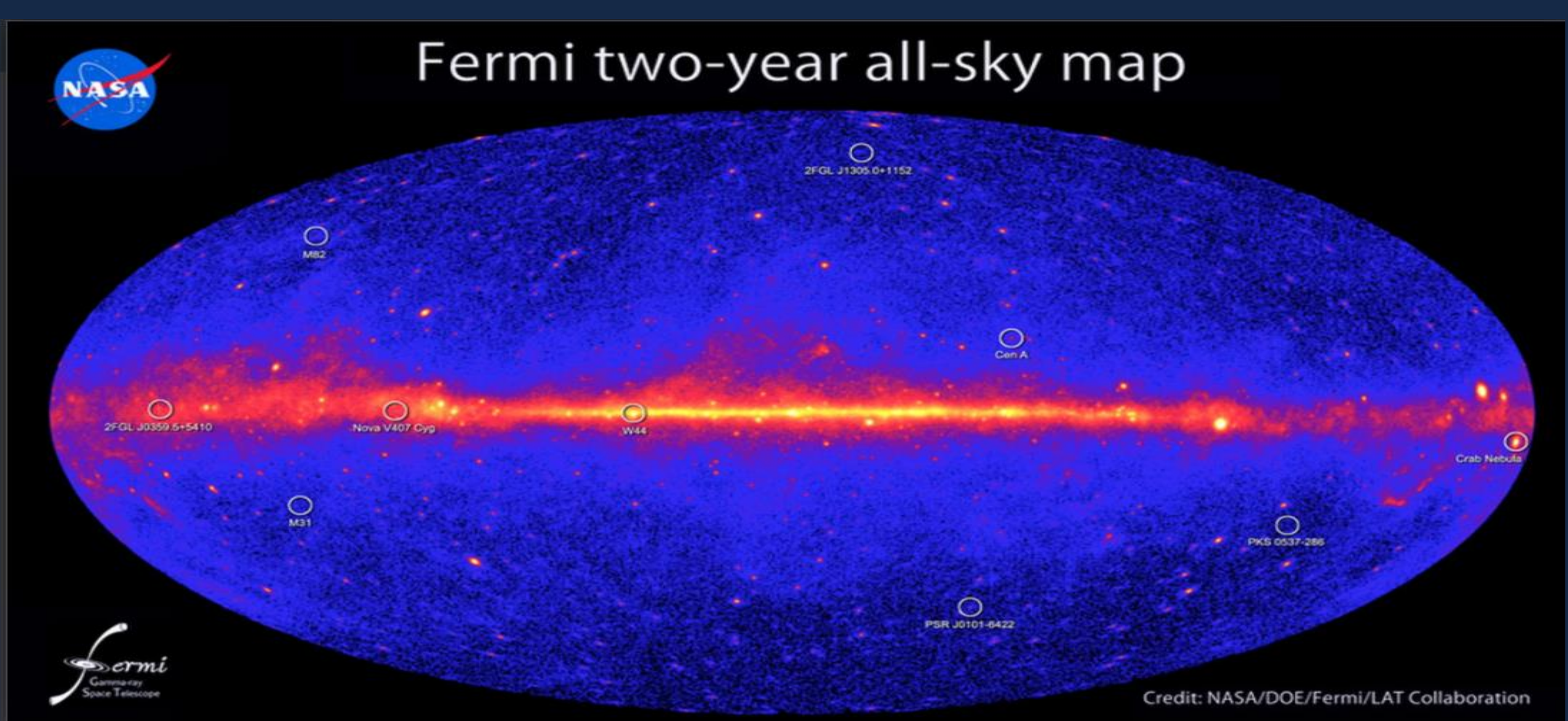
Fermi Gamma-ray Space Telescope, formerly GLAST(Gamma-ray Large Area Space Telescope) provides a better understanding of the conditions of the distant universe. Thence, Astrophysicists can study and observe the phases of evolution of the Universe. Fermi collects data by using Gamma-ray Burst Monitor (GBM) which extends the energy ranges in which bursts are observed ("Fermi: Science - Instruments - GBM," 2011). Thus, gamma-rays can be detected between 8keV to 40 MeV("Fermi Gamma-Ray Burst Monitor," n.d.) .

Fermi Gamma-ray Space Telescope has collected a detailed large data, which includes photon energies and arrival times with 2-micro sec resolution, for more than 10 years. In this project, we aim to analyze and make some observations on the data. To make the analyzing process faster and smoother, we develop algorithms to manage the database and to identify possible inconsistencies in data to prevent glitches in the automated search by using the Python programming language.

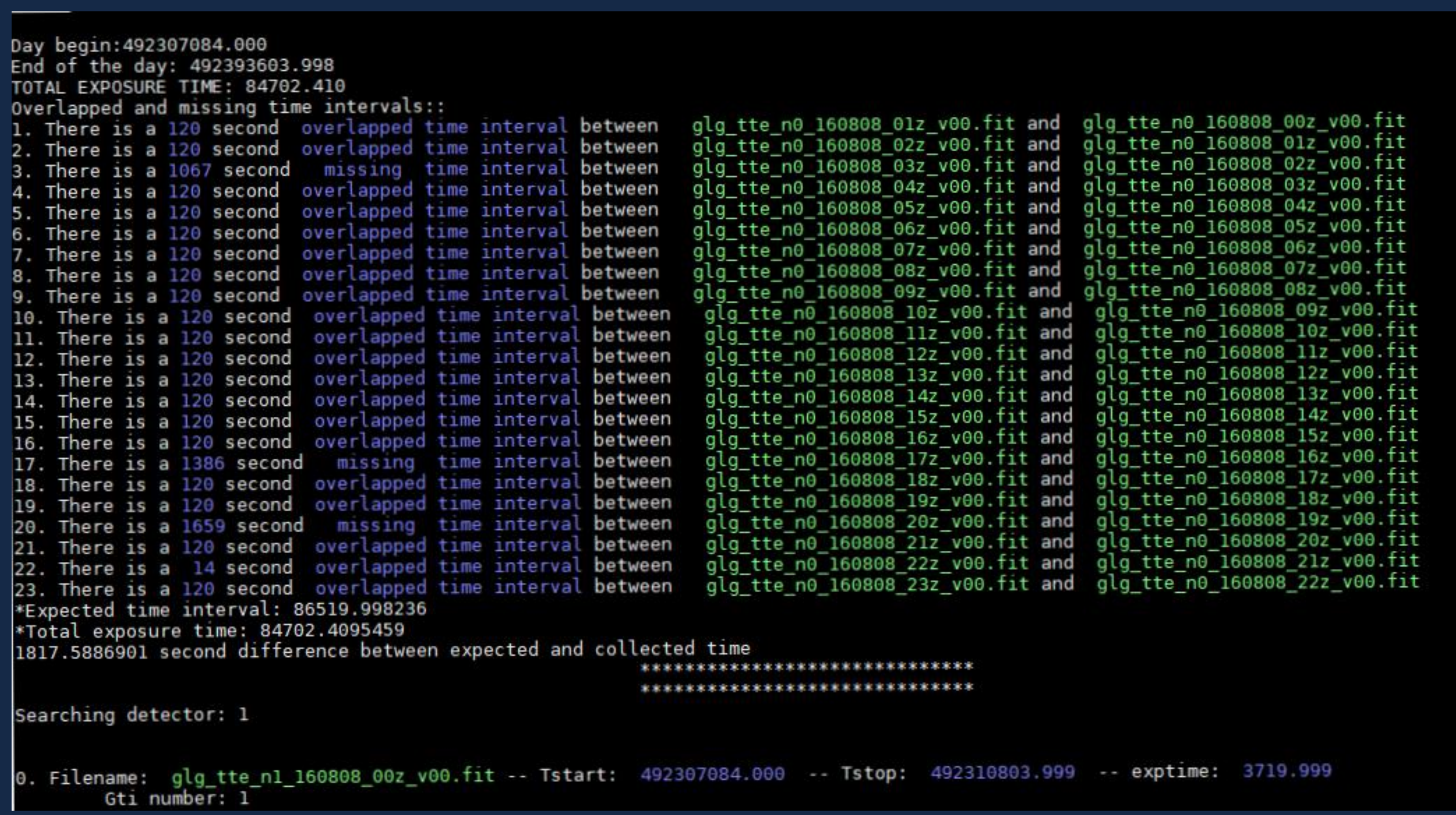
OBJECTIVES

To develop a searching and filtering algorithm by using the Python programming language in order to make analyzing process of Gamma-ray transient events faster in Fermi's GBM data.

METHODS & PROCESSES



Fermi Gamma-ray Burst Monitor's data is available on NASA's website. Therefore, our supervisors' downloaded the necessary data. In the first weeks; before writing a Python code related to our project, we learned and studied the contents of the data to understand what will be needed in the filtering algorithm. In addition to that, we are using a Linux terminal in our project. Therefore, we needed to learn basic command for Linux in the first weeks.

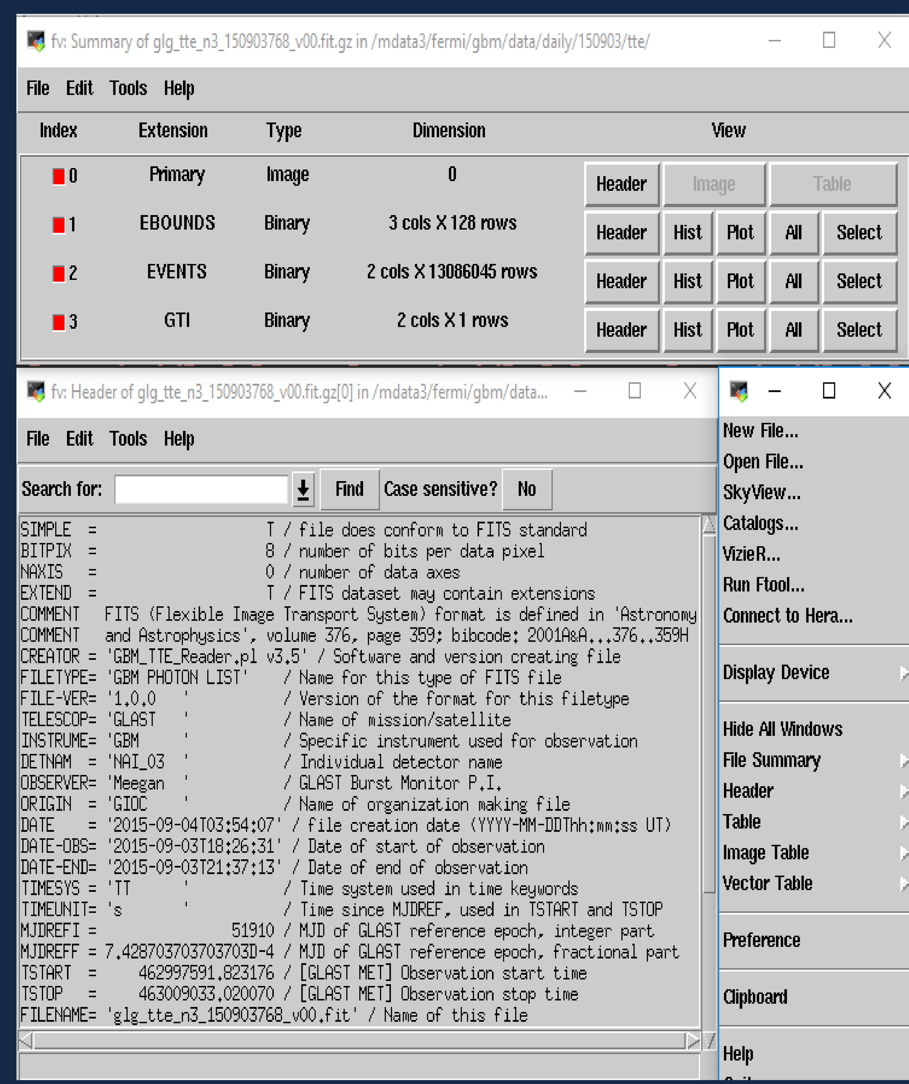


```
Day begin:492307084.000
End of the day: 492393603.998
TOTAL EXPOSURE TIME: 84702.410
Overlapped and missing time intervals::
1. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_01z_v00.fit and glg_tte_n0_160808_00z_v00.fit
2. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_02z_v00.fit and glg_tte_n0_160808_01z_v00.fit
3. There is a 1067 second   missing  time interval between  glg_tte_n0_160808_03z_v00.fit and glg_tte_n0_160808_02z_v00.fit
4. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_04z_v00.fit and glg_tte_n0_160808_03z_v00.fit
5. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_05z_v00.fit and glg_tte_n0_160808_04z_v00.fit
6. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_06z_v00.fit and glg_tte_n0_160808_05z_v00.fit
7. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_07z_v00.fit and glg_tte_n0_160808_06z_v00.fit
8. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_08z_v00.fit and glg_tte_n0_160808_07z_v00.fit
9. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_09z_v00.fit and glg_tte_n0_160808_08z_v00.fit
10. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_10z_v00.fit and glg_tte_n0_160808_09z_v00.fit
11. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_11z_v00.fit and glg_tte_n0_160808_10z_v00.fit
12. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_12z_v00.fit and glg_tte_n0_160808_11z_v00.fit
13. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_13z_v00.fit and glg_tte_n0_160808_12z_v00.fit
14. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_14z_v00.fit and glg_tte_n0_160808_13z_v00.fit
15. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_15z_v00.fit and glg_tte_n0_160808_14z_v00.fit
16. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_16z_v00.fit and glg_tte_n0_160808_15z_v00.fit
17. There is a 1386 second   missing  time interval between  glg_tte_n0_160808_17z_v00.fit and glg_tte_n0_160808_16z_v00.fit
18. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_18z_v00.fit and glg_tte_n0_160808_17z_v00.fit
19. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_19z_v00.fit and glg_tte_n0_160808_18z_v00.fit
20. There is a 1659 second   missing  time interval between  glg_tte_n0_160808_20z_v00.fit and glg_tte_n0_160808_19z_v00.fit
21. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_21z_v00.fit and glg_tte_n0_160808_20z_v00.fit
22. There is a  14 second  overlapped time interval between  glg_tte_n0_160808_22z_v00.fit and glg_tte_n0_160808_21z_v00.fit
23. There is a 120 second  overlapped time interval between  glg_tte_n0_160808_23z_v00.fit and glg_tte_n0_160808_22z_v00.fit
*Expected time interval: 86519.998236
*Total exposure time: 84702.4095459
1817.5886901 second difference between expected and collected time
                                        ******************************
                                        ******************************

Searching detector: 1

0. Filename:  glg_tte_n1_160808_00z_v00.fit -- Tstart:  492307084.000  -- Tstop:  492310803.999  -- exptime:  3719.999
       Gti number: 1
```

After learning the basic knowledge about Linux and data, We started to write a Python code to get the header, for those unfamiliar with headers, they consist of a list of 80 byte "cards", where a card contains a keyword, a value, and a comment, of the file. For that, we import Astropy library to make calculations and reading Flexible Image Transport System (FITS), which is the data format that is commonly used in astronomy field, files ("FITS File handling (astropy.io.fits) — Astropy v3.2.1," 2019). To develop our code, we tried to automate our code to read all the detector of a one day since we have 12 detectors on GBM. Then we tried to automate our code for every file that we currently have in our project.



After recent developments on the data by Nasa, which was an update for the file, we have rewritten our code to work on the updated version. To clarify, the update was related to time intervals in the data for a one day for each detector. In the new version, a single day data for a single detector was divided into 24. So, there are 12 x 24 = 288 files per day. Furthermore, we made visual improvements to our output by adding a different color to make the reading process smoother.

To conclude, we made an algorithm that search the data to reach exposure time of each day. We use input the specify the date or we can automatically run the code to reach every file's exposure time for every day have in the data to detect overlap and missing time intervals. An example can be of the output can be seen above.

PLANS FOR THE PROJECT

In the next stage, the comparison of GTI (Good Time Interval) and the difference between TStart and TStop will be done and, the plotting of the updated data will be covered. Then, using histogram will be the next step for the process. Moreover, reading the actual data instead of the header of a file will be the next challenging part, that will be studied in this project.

To summarize, improved data management and upgraded processing tools should be prepared in order to have an advancement in the project

REFERENCES

- The Fermi Gamma-ray Space Telescope. (2019, May 9). Retrieved from https://fermi.gsfc.nasa.gov/#whatsfermi
- Fermi: Science - Instruments - GBM. (2011, November 1). Retrieved from https://fermi.gsfc.nasa.gov/science/instruments/gbm.html
- Fermi GBM. (n.d.). Retrieved from https://www.mpe.mpg.de/617954/Fermi-GBM
- FITS File handling (astropy.io.fits) — Astropy v3.2.1. (2019, June 16). Retrieved from https://docs.astropy.org/en/stable/io/fits/