February 21, 2021

In this part of the Jupyter Notebook, I will be using Python 3 to analyze the provided file. I'll be making use of a few data analysis and statistics libraries and packages to do this, namely Pandas, Numpy, and Matplotlib.

[81]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from datetime import datetime
```

Here I will be loading the .csv file into a Pandas dataframe. This will put it into a format that is much easier to investigate and analyze.

[82]:
```
df = pd.read_csv('Downloads/sat_realtime_telemetry.csv')
```

I will now use the .head() method to take a look at the first five rows of the dataset, just to get a feel for it.

[83]:
```
df.head()
```

[83]:
```
  Satellite Date/Time UTC  Solar Panel Voltage X mV  Solar Panel Voltage Y mV  \
0   2019-03-21 17:45:26.0                      4388                      4270
1   2019-03-21 17:45:31.0                      4404                      4505
2   2019-03-21 17:45:36.0                      4304                      4410
3   2019-03-21 17:45:41.0                      4444                      4427
4   2019-03-21 17:45:46.0                      4505                      4444

   Solar Panel Voltage Z mV  Total Photo Current mA  Battery Voltage mV  \
0                      4198                     187                8292
1                      4186                     186                8292
2                      4175                     164                8292
3                      4164                     194                8292
4                      4460                     147                8292

   Total System Current mA  Boost Converter Temp 1 C  \
0                      141                        22
1                      146                        22
```

```
2                    151                    22
3                    146                    22
4                    146                    22

   Boost Converter Temp 2 C  Boost Converter Temp 3 C  ...  \
0                        22                        22  ...
1                        22                        22  ...
2                        22                        22  ...
3                        22                        22  ...
4                        22                        22  ...

   3.3 Bus Current mA  5.0 Bus Voltage mV  RF Temperature C  \
0               144.0              4956.0             25.70
1               134.0              4956.0             24.84
2               144.0              4956.0             24.84
3               133.0              4956.0             25.70
4               141.0              4956.0             24.84

   Receive Current mA  RF Current 3.3V Bus mA  RF Current 5.0V Bus mA  \
0                  41                      58                      29
1                  41                      59                      29
2                  41                      58                      29
3                  41                      59                      29
4                  41                      58                      29

   PA Device Temperature C  PA Bus Current mA  Antenna Temp 0 C  \
0                     34.9               38.3              15.3
1                     34.9               41.6              16.5
2                     34.9               38.8              16.5
3                     34.9               41.6              15.3
4                     34.9               38.8              16.5

   Antenna Temp 1 C
0              15.3
1              15.3
2              15.3
3              15.3
4              15.3

[5 rows x 29 columns]
```

Note that the Pandas library truncates the dataframe if the number of columns exceed 20. For this reason, I will print out all the columns so I can see which ones are hidden.

[84]:
```
df.columns
```

[84]:
```
Index(['Satellite Date/Time UTC', 'Solar Panel Voltage X mV',
       'Solar Panel Voltage Y mV', 'Solar Panel Voltage Z mV',
       'Total Photo Current mA', 'Battery Voltage mV',
       'Total System Current mA', 'Boost Converter Temp 1 C',
       'Boost Converter Temp 2 C', 'Boost Converter Temp 3 C',
       'Battery Temp C', 'Sun Sensor X+', 'Sun Sensor Y+', 'Sun Sensor Z+',
       'Solar Panel Temp X+ C', 'Solar Panel Temp X- C',
       'Solar Panel Temp Y+ C', 'Solar Panel Temp Y- C', '3.3 Bus Voltage mV',
       '3.3 Bus Current mA', '5.0 Bus Voltage mV', 'RF Temperature C',
       'Receive Current mA', 'RF Current 3.3V Bus mA',
       'RF Current 5.0V Bus mA', 'PA Device Temperature C',
       'PA Bus Current mA', 'Antenna Temp 0 C', 'Antenna Temp 1 C'],
      dtype='object')
```

This project somewhat reminds me of my experience with using NASA's TESS telescope data to locate exoplanets in other star systems. In that project, I used a Python package named Lightkurve to analyze the pixel files of stars captured by TESS and employed the popular "transit method" to find dips in the luminosity vs time graph of the star. If there was a dip in the graph, it suggested that there was something between the telescope and the star at that instant. A periodic dip meant a possible sign of an exoplanet orbiting the star.

For this project, I thought I'd use a similar approach by plotting the voltages of the solar panels, the voltages of the batteries, and the total photovoltaic current of the system against time. I reasoned that values would spike as the spacecraft neared the equator on the light side of Earth, and recede as the craft passed over the poles over to the dark side of the planet. If these graphs were periodic in nature, I could then use the formulas for orbital speed to calculate the height of the spacecraft above Earth's surface, as well as its velocity.

First, I need to ascertain the data types of the values I will be plotting, and whether any typecasting will be required.

[85]:
```
df.dtypes
```

[85]:
```
Satellite Date/Time UTC      object
Solar Panel Voltage X mV      int64
Solar Panel Voltage Y mV      int64
Solar Panel Voltage Z mV      int64
Total Photo Current mA        int64
Battery Voltage mV            int64
Total System Current mA       int64
Boost Converter Temp 1 C      int64
Boost Converter Temp 2 C      int64
Boost Converter Temp 3 C      int64
Battery Temp C                int64
Sun Sensor X+               float64
Sun Sensor Y+               float64
Sun Sensor Z+               float64
Solar Panel Temp X+ C       float64
Solar Panel Temp X- C       float64
```

```
Solar Panel Temp Y+ C        float64
Solar Panel Temp Y- C        float64
3.3 Bus Voltage mV           float64
3.3 Bus Current mA           float64
5.0 Bus Voltage mV           float64
RF Temperature C             float64
Receive Current mA             int64
RF Current 3.3V Bus mA         int64
RF Current 5.0V Bus mA         int64
PA Device Temperature C      float64
PA Bus Current mA            float64
Antenna Temp 0 C             float64
Antenna Temp 1 C             float64
dtype: object
```

The 'Satellite Date/Time UTC' column is of type 'object', which means I will have to convert it into a data type that can be parsed by the graphing libraries. All of the other columns are either integers or floating-point values, so they'll work just fine as they are.

[86]:
```
df['Satellite Date/Time UTC'] = pd.to_datetime(df['Satellite Date/Time UTC'])
```

Let's check to see the data type of this column now:

[87]:
```
df['Satellite Date/Time UTC'].dtype
```

[87]:
```
dtype('<M8[ns]')
```

The values are now formatted as Python 'datetime' types.

For the sake of convenience, lets rename our columns of interest to a more standard format, without spaces or capital letters.

[88]:
```
df.rename(columns = {'Satellite Date/Time UTC' : 'datetime', 'Total Photo␣
↪Current mA':'total_current', 'Battery Voltage mV' : 'battery_voltage', 'Solar␣
↪Panel Voltage X mV': 'solar_voltage'}, inplace = True)
```

Let's take a look at our dataset now:

[89]:
```
df.head()
```

[89]:
```
             datetime  solar_voltage  Solar Panel Voltage Y mV  \
0 2019-03-21 17:45:26           4388                      4270
1 2019-03-21 17:45:31           4404                      4505
2 2019-03-21 17:45:36           4304                      4410
3 2019-03-21 17:45:41           4444                      4427
4 2019-03-21 17:45:46           4505                      4444

   Solar Panel Voltage Z mV  total_current  battery_voltage  \
0                      4198            187             8292
1                      4186            186             8292
```

```
2                     4175             164              8292
3                     4164             194              8292
4                     4460             147              8292

   Total System Current mA  Boost Converter Temp 1 C  \
0                      141                        22
1                      146                        22
2                      151                        22
3                      146                        22
4                      146                        22

   Boost Converter Temp 2 C  Boost Converter Temp 3 C  ...  \
0                        22                        22  ...
1                        22                        22  ...
2                        22                        22  ...
3                        22                        22  ...
4                        22                        22  ...

   3.3 Bus Current mA  5.0 Bus Voltage mV  RF Temperature C  \
0               144.0              4956.0             25.70
1               134.0              4956.0             24.84
2               144.0              4956.0             24.84
3               133.0              4956.0             25.70
4               141.0              4956.0             24.84

   Receive Current mA  RF Current 3.3V Bus mA  RF Current 5.0V Bus mA  \
0                  41                      58                      29
1                  41                      59                      29
2                  41                      58                      29
3                  41                      59                      29
4                  41                      58                      29

   PA Device Temperature C  PA Bus Current mA  Antenna Temp 0 C  \
0                     34.9               38.3              15.3
1                     34.9               41.6              16.5
2                     34.9               38.8              16.5
3                     34.9               41.6              15.3
4                     34.9               38.8              16.5

   Antenna Temp 1 C
0              15.3
1              15.3
2              15.3
3              15.3
4              15.3

[5 rows x 29 columns]
```

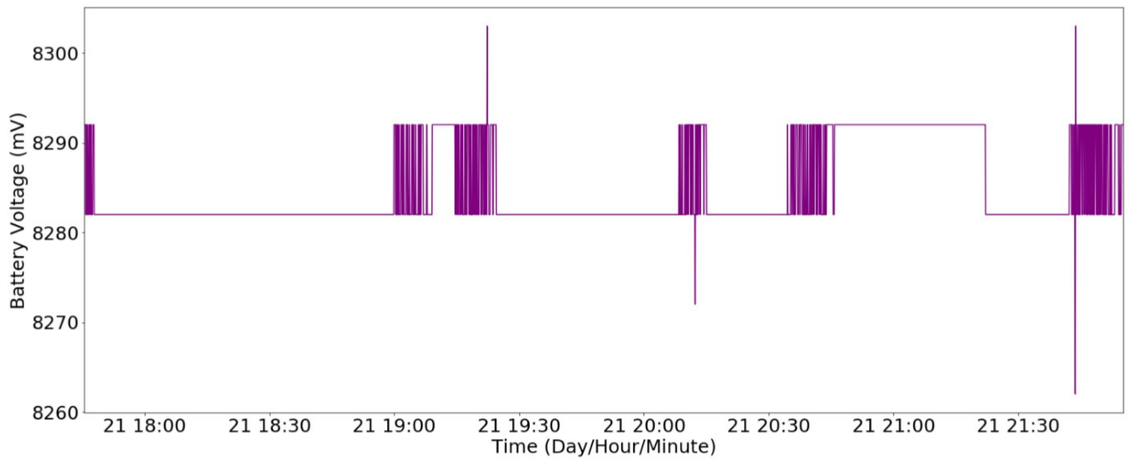
Let's now plot the battery voltage against time.

[90]:
```
fig, ax = plt.subplots(figsize=(25, 10))

# Add x-axis and y-axis
ax.plot(df['datetime'],
        df['battery_voltage'],
        color='purple')

# Set title and labels for axes
ax.set_xlim(df['datetime'].min(), df['datetime'].max())
ax.set_xlabel('Time (Day/Hour/Minute)', fontsize = 25)
ax.set_ylabel('Battery Voltage (mV)', fontsize = 25)
ax.tick_params(axis='both', which='major', labelsize=25)

plt.show()
```



There seems to be a somewhat periodic nature to this graph. For a craft to be in Low Earth Orbit, it generally must have an orbital period of less than 128 minutes. For reference, the ISS has an orbital period of about 92.7 minutes at a mean altitude of 400km above the surface of the Earth.

Let's now plot the solar panel voltage, as well as the total photo current, against time, and see if that gives us any more information.

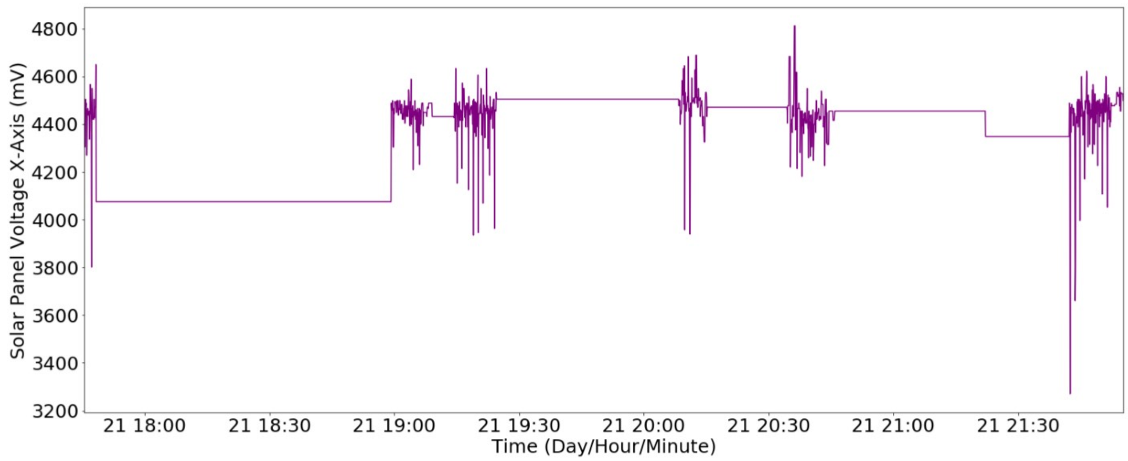
[91]:
```
fig, ax = plt.subplots(figsize=(25, 10))

# Add x-axis and y-axis
ax.plot(df['datetime'],
        df['solar_voltage'],
        color='purple')

# Set title and labels for axes
```

```
ax.set_xlim(df['datetime'].min(), df['datetime'].max())
ax.set_xlabel('Time (Day/Hour/Minute)', fontsize = 25)
ax.set_ylabel('Solar Panel Voltage X-Axis (mV)', fontsize = 25)
ax.tick_params(axis='both', which='major', labelsize=25)

plt.show()
```

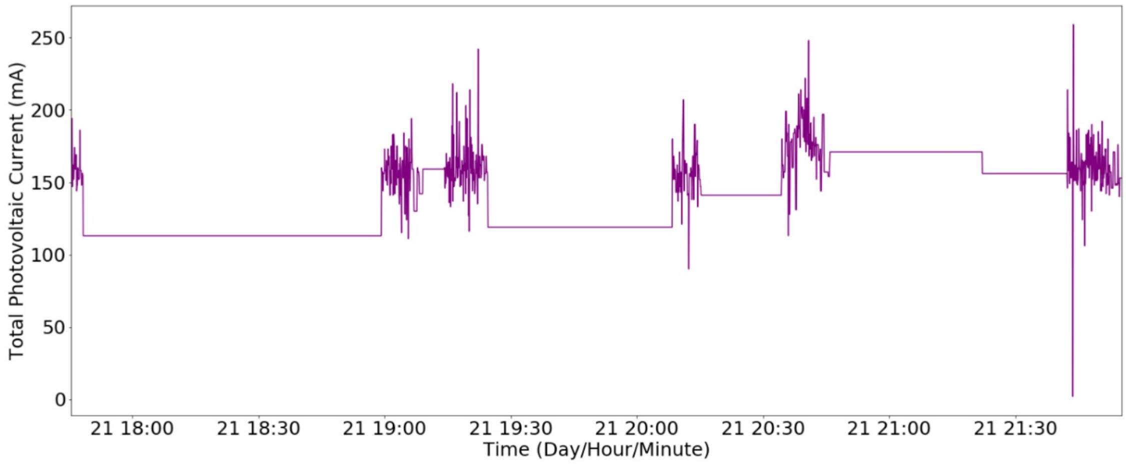


[92]:
```
fig, ax = plt.subplots(figsize=(25, 10))

# Add x-axis and y-axis
ax.plot(df['datetime'],
        df['total_current'],
        color='purple')

# Set title and labels for axes
ax.set_xlim(df['datetime'].min(), df['datetime'].max())
ax.set_xlabel('Time (Day/Hour/Minute)', fontsize = 25)
ax.set_ylabel('Total Photovoltaic Current (mA)', fontsize = 25)
ax.tick_params(axis='both', which='major', labelsize=25)

plt.show()
```



All of these graphs seem to display a spike that starts at around 19:00 hours. My guess is that this is when the craft emerges from the dark side of the Earth and the solar cells on the craft are exposed to sunlight. This is likely when the batteries enter a charging state. A second, larger spike occurs about 20 minutes later, suggesting that the craft has passed the equator at that instant and the solar cells are receiving maximal exposure to sunlight. The spikes and depressions may also be due to the pivoting of the solar panels to be on-or-off point with regard to alignment towards the sun.

Looking at the plots, these patterns seem to repeat every 77-80 minutes or so, although due to the fact that the dataset only encompasses about 4 hours and 10 minutes of flight time, this is a bit tough to accurately predict.

This is quite a fast orbital period indeed, which leads me to believe that this spacecraft is in a powered orbit around the Earth, rather than a purely gravitational one. This means that Kepler's Laws and the formulas for orbital speed wouldn't really apply.