Assignment - Optimization Engineer (Computation)

Welcome to the world of space and satellites, where we track satellite positions using TLEs (https://en.wikipedia.org/wiki/Two-line_element_set). Please find the attached file containing information of about 30 satellites in TLE format - [sample file - ***30sats.txt***].

In this assignment you need to create a program that does the tasks below as explained.

Step 1. Generate the state vectors of the satellite

To find the location of a satellite in space, you can use the SGP4 propagator and the https://pypi.org/project/sgp4/ library. Simply go through the documentation to understand how to use it.

This library contains functions to retrieve the location of a satellite using TLE data. We have attached a text file containing TLEs for you to use.

Retrieve the position of each satellite for five days using a ***one second*** time interval. Obtain the data in the following format: ***time, P(x), P(y), P(z), V(x), V(y), V(z),*** where ***P*** represents position and ***V*** represents velocity.

Step 2. Convert data to [Lat, Long, Alt] format

```
def ecef2lla(i, pos_x, pos_y, pos_z):
    ecef = pyproj.Proj(proj="geocent", ellps="WGS84", datum="WGS84")
    lla = pyproj.Proj(proj="latlong", ellps="WGS84", datum="WGS84")
    lona, lata, alta = pyproj.transform(
        ecef, lla, pos_x[i], pos_y[i], pos_z[i], radians=False)
    return lona, lata, alta
```

NOTE: pyproj is a library from which you will be using only the function ‘ecef2lla’.

The above function takes lists Px, Py, and Pz as input in ***list*** format and returns longitude, latitude, and altitude.

Let's call this result LLA:

Step 3. Find when the satellite passes over a chosen lat long region.

Now that you have converted the location from XYZ to latitude, longitude, and altitude, ask the user for four coordinates (rectangle).

Filter the results from LLA to only include outputs that are located between user-defined locations.

for example :

Latitude: 16.66673, Longitude: 103.58196

Latitude: 69.74973, Longitude: -120.64459

Latitude: -21.09096, Longitude: -119.71009

Latitude: -31.32309, Longitude: -147.79778

Assume that there will be multiple regions (different combinations of user defined locations).

Step 4. Optimize the code to reduce computation time

- This program has to be scaled to ingest about 27,000 satellites [sample file - 27000sats.txt] and to analyse using a time-step of 0.1 second.
- Optimise performance of the code: RAM, CPU/GPU, computation time and so on.
- Use distributed computing to exponentially reduce the computation time.
- Prepare modularized production ready code, you can code in your preferred programming language(s).

What we are looking for:

- We will primarily be looking at your performance in section 4 (optimising for resources & time).
- Good documentation is a plus.
- Solutions completely provided by online tools are not recommended and will be highly penalised as an evaluation criteria.

Submission

- Please share your solution as a git repository with a readme file for project setup.
- Add one or two page explanations on how the code is structured, along with a **SUMMARY TABLE presenting the computational time and effort** (RAM, CPU, GPU etc.).