Week 3 Report

Dhanush Balusa

June 4, 2025

Research

Coordinate System: True Equator, Mean Equinox (TEME)

- True Equator: The coordinate plane uses the actual orientation of Earth's equator at a given time.
- Mean Equinox: The reference for the vernal equinox does not include short-period variations like nutation (periodic "wobbling" motion of Earth's rotation axis caused by the gravitational influence of the Moon and Sun on Earth's equatorial bulge).
- https://astronomy.stackexchange.com/questions/44140/can-someone-explain-to-me-the-t

SGP4

- Note: Last week I learned that TLEs are not accurate for long periods of time and that they are updated frequently.
- https://conference.sdo.esoc.esa.int/proceedings/sdc6/paper/41/SDC6-paper41.pdf

Other resources:

- https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/pdf/individual_docs/17_frames_and_coordinate_systems.pdf

Previous Code

My code from the previous week calculated the approach conditions between the ISS and NOAA 15.

```python
# Define satellites
satellites_info = [
    (25544, "ISS (ZARYA)"),
    (25338, "NOAA 15"),
    (33591, "NOAA 19"),
    (22236, "COSMOS 2221")
]

# Load timescale
ts = load.timescale()

# Fetch and create satellites
satellites = []
for norad_id, name in satellites_info:
    sat = fetch_and_create_satellite(norad_id, name, username, password)
    satellites.append(sat)


# Time range: 24 hours, 1-hour steps
#start_time = satellites[0].epoch
start_time = ts.utc(2024, 2, 27, 12, 0, 0)  # a UTC time
time_steps = ts.utc(start_time.utc.year, start_time.utc.month, start_time.utc.day, range(0, 48))

# Store results
times = []
relative_positions = {name: [] for _, name in satellites_info[1:]}
approach_speeds = {name: [] for _, name in satellites_info[1:]}

for t in time_steps:
    states = [sat.at(t) for sat in satellites]
    iss_state = states[0]
    for i, state in enumerate(states[1:], start=1):
        rel_pos = iss_state.position.km - state.position.km
        rel_vel = iss_state.velocity.km_per_s - state.velocity.km_per_s
        approach_speed = np.linalg.norm(rel_vel)
        relative_positions[satellites_info[i][1]].append(np.linalg.norm(rel_pos))
        approach_speeds[satellites_info[i][1]].append(approach_speed)
    times.append(t.utc_iso())
```

Figure 1: Code from week 2 calculating the approach conditions between ISS and a couple of other satellites

Code Changes

Based on the feedback from last week's team meeting, I decided to make the following changes:

1. **My first change was to check if the TLE data is fetched correctly.**

   Observation: All the TLE data is fetched correctly, it is the most recent TLE data (as of 5/28/2025).

```
0 ISS (ZARYA)
1 25544U 98067A   25155.13777713  .00014923  00000-0  26896-3 0  9997
2 25544  51.6376  11.7370 0001938 182.3055 323.8206 15.49999521513135
```

Figure 2: TLE data from Space Track for ISS (ZARYA)

```
ISS (ZARYA) TLE:
1 25544U 98067A   25155.13777713  .00014923  00000-0  26896-3 0  9997
2 25544  51.6376  11.7370 0001938 182.3055 323.8206 15.49999521513135
```

Figure 3: TLE data fetched from Space Track's API for ISS (ZARYA)

2. **My second change was to fetch historical TLE data.**

```
# Function to fetch TLE data from space-track.org
def fetch_tle(norad_id, username, password, start_date, end_date):
    """
    Fetch the latest TLE data for a given NORAD Catalog ID from space-track.org.
    Returns a list of TLE lines.clea
    """
    LOGIN_URL = 'https://www.space-track.org/ajaxauth/login'
    TLE_URL = (
    f'https://www.space-track.org/basicspacedata/query/class/tle/'
    f'NORAD_CAT_ID/{norad_id}/EPOCH/{start_date}--{end_date}/orderby/EPOCH%20desc/format/tle'
)
```

Figure 4: I updated the fetch_tle functiion to get historical TLE data

   Observation: It seems like the code fetches the TLE data correctly.

3. **My third change was to compare the approach conditions of the TIMED satellite and Cosmos 2221.**

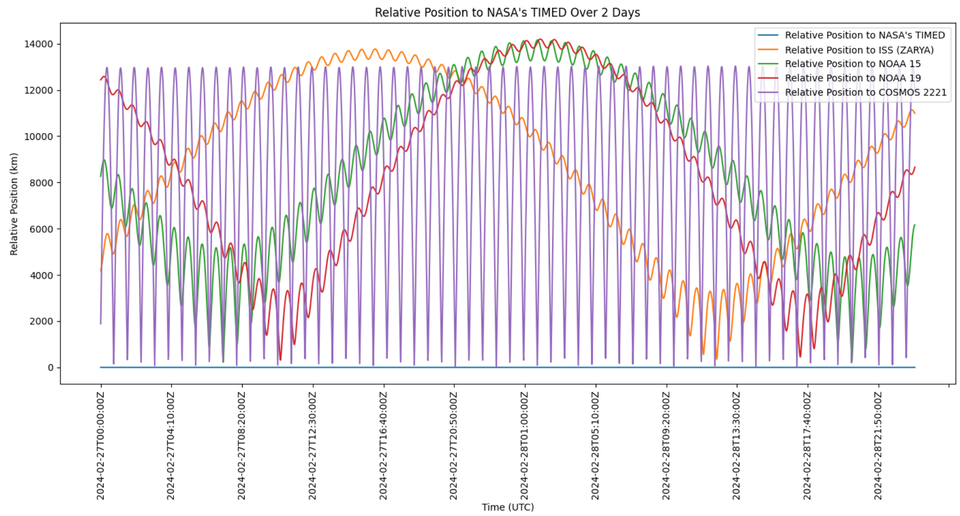

Figure 5: NASA's TIMED satellite's near miss with Cosmos 2221

Note: Cosmos 2221 is the odd satellite out, why? Why does it follow a sinusodal patern in so quickly? If I try using GMAT with TLEs that could help me.

```
Near Miss Detected Between TIMED and COSMOS 2221:
   - Closest Distance:        24.338 km
   - Approach Speed:          14.126 km/s
   - Time (UTC):              2024-02-28T17:02:00Z
```

Figure 6: Approach conditions of TIMED satellite and Cosmos 2221.

Issue: The correct time is around 6:30 UTC on February 28, 2024. The approach speed is fluctuating too much, I need to check the code for errors.

4. **My fourth change was adding Iridium 33's collision with Cosmos 2251 from 2009.**

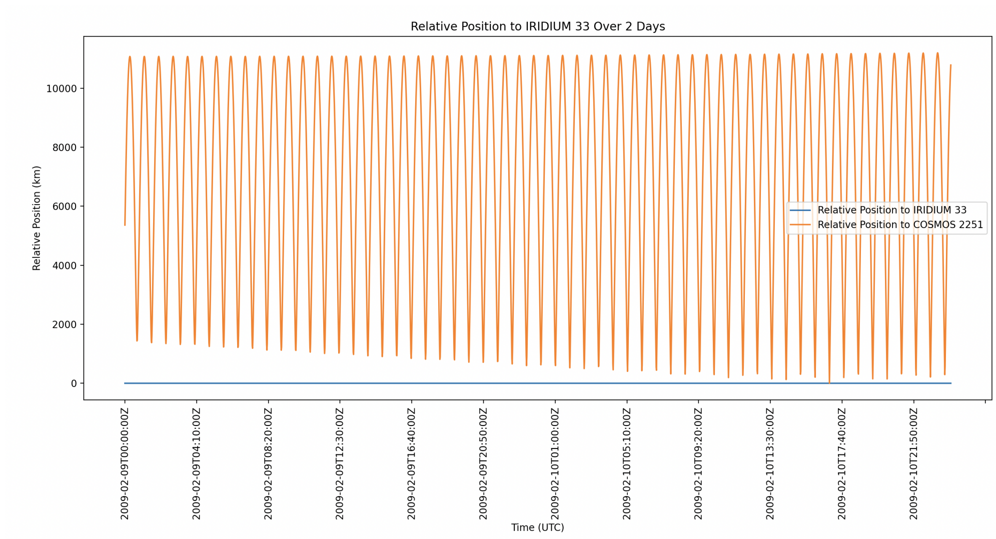

Figure 7: Iridium 33 collides with Cosmos 2251 around 16:56 UTC on February 10, 2009.

```
Near Miss Detected Between Iridium 33 and COSMOS 2251:
  - Closest Distance:       2.372 km
  - Approach Speed:         11.647 km/s
  - Time (UTC):             2009-02-10T16:56:00Z
```

Figure 8: Approach conditions of Iridium 33 collision with Cosmos 2251.

Observation: The code accurately calculates the time of the collision.
Next steps: I need to calculate and cross check the approach speed and angle of the collision. Since this test case works I need to utilize it too test all my calculations. Also I need to simulate this collision using GMAT and see how accurate my calculations were.
Issue: Why can't I get the TIMED satellite and Cosmos 2221 to work?
Note: The time matches but the approach speed has to be checked.

`2009collision:https://en.wikipedia.org/wiki/2009_satellite_collision`

Next Steps

- Read more about the error in SGP4.
- Research if the compact sinusoid patern is correct. I don't understand why the relative position is fluctuating so quickly.
- Calculate the angle of approach, cross check with GMAT. Check the approach speed too.
- Loop through all the TLEs in the file and calculate the approach conditions for each satellite if it is within a certain distance (e.g., 100 km) of the ISS.
- Make my code a usable function so Catherine can just call it. Input? → Output (angle and approach speed)