Space Flight Operations
Assignment #4 - revised

AE 4361
Prof. G. Lightsey

8 February 2022

Due: 17 February 2022, 11:59 pm, submit as a pdf file on Canvas web site.

Late Submittal: Before 19 February 2022, 11:59 pm for 50% credit.

Office Hours: Professor Lightsey: Wednesday, 3:00-4:00 pm, BlueJeans (access from Canvas)
TAs: Monday, 5:00-6:00, BlueJeans (access from Canvas)
Or contact email for an appointment (see syllabus for email addresses)

Reading: Basics of Space Flight web site:
Chapter 2 (https://solarsystem.nasa.gov/basics/chapter2-1, all pages),
Chapter 13 (https://solarsystem.nasa.gov/basics/chapter13-1)

Complete the reading and then complete the Multiple Choice Quiz on Canvas, which is due by February 15 at 5:00 pm.

This assignment also includes a MATLAB portion. Attach a copy of your code at the end of your assignment submission. Your code must include useful comments, name of the author, and the date of creation. Also, when necessary, include a screenshot of the code output.

Some of the problems will require you to solve Kepler's Equation, which can be done with the Newton-Raphson root finding method. For your convenience, two MATLAB scripts have been included with the assignment: ***NewtRaph.m*** and ***Kepler_Solver.m***. Feel free to use both scripts to compute your answers for the problems that follow. You will need to copy both files into the same folder. Note that the ***Kepler_Solver.m*** file calls the ***NewtRaph.m*** script, so you will only need to call ***Kepler_Solver.m*** to compute your answers. Refer to the code comments for more information on how to run the script.

Problem 1.

This assignment will build on the function created in Assignment 3. If you recall, the function that was meant to take in 5 orbital elements (a, e, i, Ω, and w), as well as the time since perigee crossing (t), and then output the position coordinates of the vehicle in the Earth-Centered Inertial (ECI) reference frame. Once again, verify that your function is working as intended with the following test case:

Input (ISS Orbit):

a = 6.796620707 x 106 m; i = 51.6439°; e = 2.404 x 10-4

Ω = 86.8571°; w = 1.8404°; t = 100 s

Output:

−0.239086091

$\boldsymbol{r}_{ISS}$ = [6.747093009] × 106 m

0.769129531

If your function output does not match the results included above, you have one of two options: you can try to fix your function (this is encouraged since it helps you learn the course material) or you can use the working version of the function provided with the assignment (the file is called ***OrbElem2Pos.m***). If you opt to use the provided function, please refer to the comments included in the file for instructions on how to use it.

For this homework you will modify the Assignment 3 function by keeping the same input arguments but modifying the outputs. The function should now use the orbital elements to compute the Earth-Centered Earth Fixed (ECEF) reference frame coordinates of the vehicle over time. These coordinates will need to be converted to the respective vehicle latitude, longitude, and altitude. The function itself should output the latitude and longitude in **degrees** and the altitude in **meters.** Hence, your function should look something like this:

$$[lat, lon, h] = \boldsymbol{Your\_Function}\ (a, e, i, Omega, w, t)$$

Just as in the previous assignment, the angles should be expressed in degrees, the distances in meters, and the time in seconds. You will be calling your function repeatedly in a loop, over a specified period of time for each tested case, in order to compute the satellite position for each timestep. The result of the loop should be *n*-by-1 vectors (where *n* is the number of timesteps) for lat, lon, and h. Once you have these vectors, you can call the provided ***GroundTrack*** function (you do not have to write this function) as follows:

$$\boldsymbol{GroundTrack}(lat, lon)$$

The ***GroundTrack*** function will return a figure with the satellite's ground track for the simulated spacecraft trajectory. No changes to the provided function are necessary. However, you need to make sure that:

- The latitude and longitude inputs are in degrees.
- The latitude and longitude vectors have the same length.
- All of the contents in the ***Homework4_Code.zip*** file have been saved in the same folder as your matlab code. The function will not work otherwise.

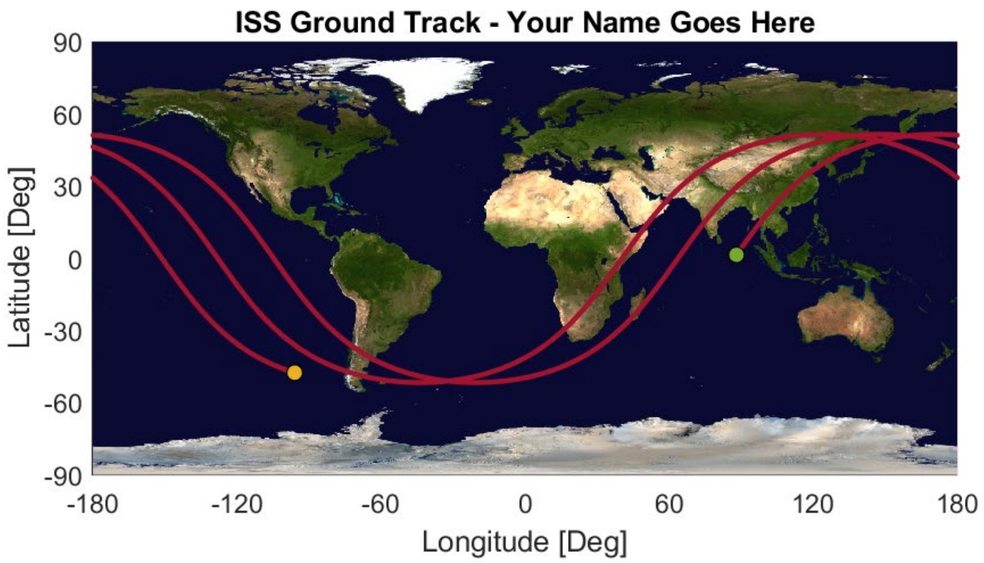
You will be generating ground track plots for several different orbits which are specified below. Your function code should not be changed for each tested case. The only parameters that should be altered are the function inputs.

Note: we will be grading your coding style as well as your answers. Good coding style and generous comments should be used to describe what is being done on each line.

You should focus first on making sure that the function you wrote is working as intended after the modifications. You can refer back to the test case given for the International Space Station (ISS). Once modified, your function should output the following results for the given orbital elements at t = 1 s :

lat = 1.4937° ; lon = 88.0353° ; h = 4.2398680 x 10 m ;

After verifying that the above result is true, run your function in a loop. You should compute the vehicle's latitude, longitude, and altitude over a span of 15,000 s at 1 s intervals. The resulting arrays can then be used as inputs for the ***GroundTrack*** function. The function output should look as follows:



The green point in the plot corresponds to the starting position of the ground track and the yellow point is the final position. You should make sure that your results match the provided figure and position output before continuing with this assignment. The remaining problems will rely on this function.

You are to make ground track plots for the following test cases. You must hand in the following deliverables for each case:

> **i.** Ground track plot for each case with your name in the title. This is to verify that the plot is your own work.
>
> **ii.** Brief description (1-2 sentences for each case) of why the ground track looks the way it does.
>
> **iii.** Your source code for the function you created. This must include name, useful comments, and date of creation. Please append the code at the end of your assignment submission.

(a) ***Sun-Synchronous Orbit - Planet Lab's FLOCK 2P-6***

$$a = 6.864718799 \times 10^6\ m; \quad i = 97.3262°; \quad e = 8.8380 \times 10^{-4};$$

$$\Omega = 218.6861°; \qquad \omega = 216.6629°; \qquad time\ span = 17{,}000s\ @\ 1s\ intervals;$$

(b) ***Molniya 2-9 Communication Satellite***

$$a = 2.323698972 \times 10^7\ m; \quad i = 64.0370°; \quad e = 0.680478;$$

$$\Omega = 343.6936°; \qquad \omega = 288.0884°; \qquad time\ span = 86{,}400s\ @\ 1s\ intervals;$$

Optional work for fun (not for credit, but it's cool!): There is an additional function that has been provided with the assignment. The function creates a 3-dimensional perspective plot of the satellite over a sphericalEarth. It looks cool and I believe it helps understand some of the ground tracks. You can try plugging in both ECEF and ECI coordinates for latitude, longitude, and altitude. The figures are not for credit but turnin these plots if you make them. You can call the function as follows:

$$\boldsymbol{Plot3D\_Earth}(Lat, Lon, h, fig)$$

Problem 2.

Planet Labs is a private Earth imaging company that operates a constellation of over 150 satellites. The constellation is mainly composed of the Dove CubeSats (pictured below):

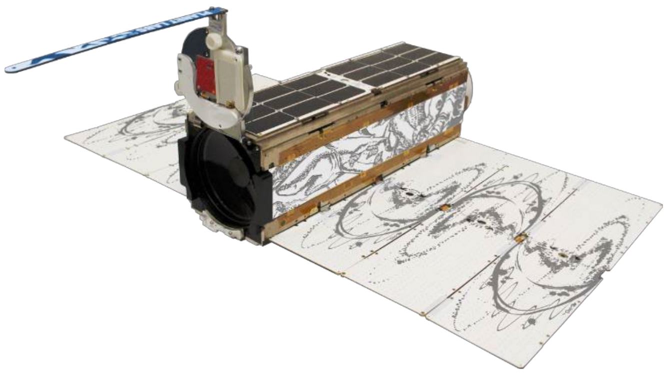


Image source: https://www.planet.com/company/approach/

The spacecraft are equipped with an Earth imaging sensor suite that allows for 3 to 5-meter resolution images of the surface. The constellations are designed in such a way as to allow for the daily imaging of the entire surface of the Earth!

One of the ground stations for tracking the satellites is located in **Brewster, WA**. The latitude (ϕ) of the ground station is 48.096° and the longitude (λ) is -119.781°. Assuming the Earth is a sphere of constant radius R_e = 6371.0 x 10^3 m,

(a) Calculate the position of the Brewster ground station in the Standard Earth-Centered Earth Fixed (ECEF) reference frame, r_{GS}, in meters rounded to the nearest meter. If you compute this using Matlab, then you must hand in a screenshot of the output of your calculations.

Answer: r_{GS} = [-2.113445   -3.693123   4.741.712]^T x 10^6 m

By repeatedly calling the function you created in Problem 1, you can create an array of n positions that describes the satellite's position versus time in the Earth-Centered Earth Fixed (ECEF) reference frame. The array should be called Γ = $Gamma$ = [$tvector$, $latvector$, $lonvector$, $hvector$] and should be an [n x 4] matrix where each of the vector arguments is an [n x 1] column vector.

Next, you will write a new Matlab function named **GSVisibilityCheck** that takes the array of satellite positions *r* from Problem 1 as an input and determines if the satellite is visible to the Planet Labs ground station located in Brewster (use your answer to part a) at each time epoch (row) in the array *r*. In order to be visible to the ground station, the satellite must be at least 20 degrees above the horizon (i.e., use an elevation mask angle of $E_O > 20°$).

The function output should be a smaller array γ = $gamma$ = [$tvector$, $latvector$, $lonvector$, $hvector$] for only those points that pass the visibility test from the ground station. The time vector should be specified in seconds, the latitude/longitude in degrees, and the altitude (h) vector in meters. In other words, your function should look as follows:

$$[gamma] = \boldsymbol{GSVisibilityCheck}(Gamma)$$

As an example output case (and to test your code), if you use the International Space Station (ISS) test case for Problem 1 every second for a total time of 65,000 seconds, then the output gamma should be a [256 x 4] matrix. The first row of that matrix is included below for reference:

$$gamma(1,:) = [6.2219 \quad 0.004202361 \quad 0.02323905 \quad 42.4725164] \times 10^4$$

The angles in the output are in degrees. It may help to plot a ground track for the satellite as you did in Problem 1 to visualize what is happening as the satellite passes over the ground station. Feel free to modify the **GroundTrack** function to highlight the regions when the satellite is visible (the ground track figures are not for credit but they look cool!).

(b) You are to use your **GSVisibilityCheck** function to compute the visibility of **Planet Lab's FLOCK 2P-6 (Problem 1 Case a)** from the Brewster ground station. Call your Problem 1 function in a loop every second for a total time of 40,000 seconds. Use the output to build the Γ = $Gamma$ array. Use this array to run your **GSVisibilityCheck** function. You must hand in the following deliverables:

i. A screenshot of the first row of the γ = $gamma$ array output from your **GSVisibilityCheck** function. Only list the first row. The screenshot must show your function call.

ii. Your source code for the function you created. This must include name, useful comments, and date of creation. Please append the code at the end of your assignment submission.

Problem 3.

Now write a Matlab function called **GSsatLOS** that takes the visible point array γ from Problem 2 as an input and transforms these points into $azimuth$ = α = azi and $elevation$ = E = ele angles expressed in the local topocentric reference frame at the ground station coordinates (i.e. at the location described by your Problem 2(a) answer). The function should look as follows:

$$[azi, ele] = \boldsymbol{GSsatLOS}(gamma)$$

The azimuth and elevation angles should be expressed in degrees.

(a) Turn in the following deliverables:

i. A screenshot of the azimuth and elevation angle outputs for the first point only of the visible γ matrix computed in Problem 2. The screenshot must show your function call.

ii. Your source code for the **GSsatLOS** function you created. This must include name, useful comments, and date of creation. Please append the code at the end of your assignment submission.

Use the built in Matlab function $\boldsymbol{polarplot}(theta, rho, 'r.')$ to make a sky plot of the azimuth and elevation angles for each visible point as a dot on a compass rose with North in the vertical-axis direction and East in the horizontal-axis direction. The center of the plot should correspond to elevation angle E = 90°. The formulas needed to help make the plot are:

$$theta = (azi) * \text{rr}/180$$
$$rho = 1 - ele/90$$

An example of the correct sky plot output for the International Space Station (ISS) test case is given on the last page. Note: In order to format the plot correctly, you must run the following commands after the call to the **polarplot** function:

```
hold on
rlim([0 1])
ax = gca;
ax.ThetaZeroLocation = 'top';
ax.ThetaDir = 'clockwise';
ax.RTickLabel = {};
```

(b) Using the $[azi, ele]$ angles you obtained in part (a), make a sky plot for the Problem 1 part (a) satellite orbit over the ground station at Brewster, WA. For each separate satellite pass (i.e. continuous sky track of a satellite), indicate on the plot (based on the data points in the plot):

i. The first point of each satellite pass and the satellite rise time, elevation, and azimuth angle.
ii. The maximum elevation angle of each satellite pass and the time of maximum elevation.
iii. The last point of each satellite pass and the satellite set time, elevation, and azimuth angle.
iv. For each satellite pass, indicate the direction of motion with an arrow on the plot
v. Write your name and date on the plot (this could be in the title), indicating that this is your own work.

Turn in an annotated printout of your annotated satellite sky plot, similar to what is provided in the example plot in the following page.