Solar Gravity Influences on Earth-Moon Three-Body Periodic Orbits

MANE 4100 Term Project Spring 2023

By Cory Puckett

Abstract

This paper investigates the effects of solar gravity on the stability of known periodic orbits in the Earth and Moon system. This will help gain a better understanding of how chaotic this system is. The known periodic orbit initial conditions are taken from NASA's Jet Propulsion Lab (JPL) Solar System Dynamics (SSD) website [1]. This website was also used to source location information for the sun relative to the Moon to calculate the effects of Solar gravity on the craft [2]. A matlab live script was made to allow a user to pick almost any one of the orbits given on the SSD website and see how the solar gravity affects them.

Introduction

In any two body system there exist 5 Lagrange points which are points where if a point mass is added with mass substantially smaller than the two other bodies it will stay stationary relative to those two other bodies. These Lagrange points can be seen in figure 1.

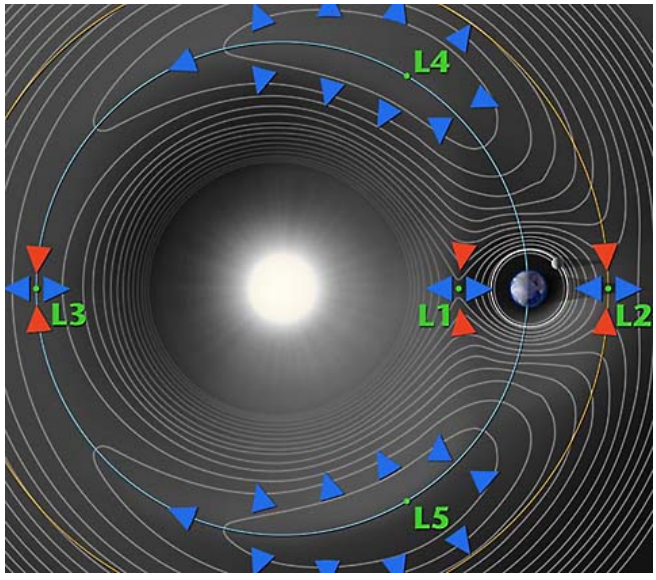


Figure 1: Lines of potential in the Earth Sun system with the Lagrange points labeled. [3]

As seen in figure 1 the first three Lagrange points are in line with the two primary bodies and are all unstable. L4 and L5 can sometimes be stable depending on the mass ratio of the primary and secondary bodies. Many of the two body systems in our solar systems have stable L4 and L5 points, such as the Earth-Moon, Sun-Earth, and Sun-Jupiter systems. Though the first 3 points are unstable, periodic orbits near them can be found. NASA's Jet Propulsion Lab (JPL) Solar System Dynamics (SSD) [1] website includes a very large database of tens of thousands of periodic orbits for different two body systems including the Earth-Moon system which was chosen to be focused on in this paper. This database includes a large variety of different styles of orbits. Some of these such orbit families can be seen in figure 2.

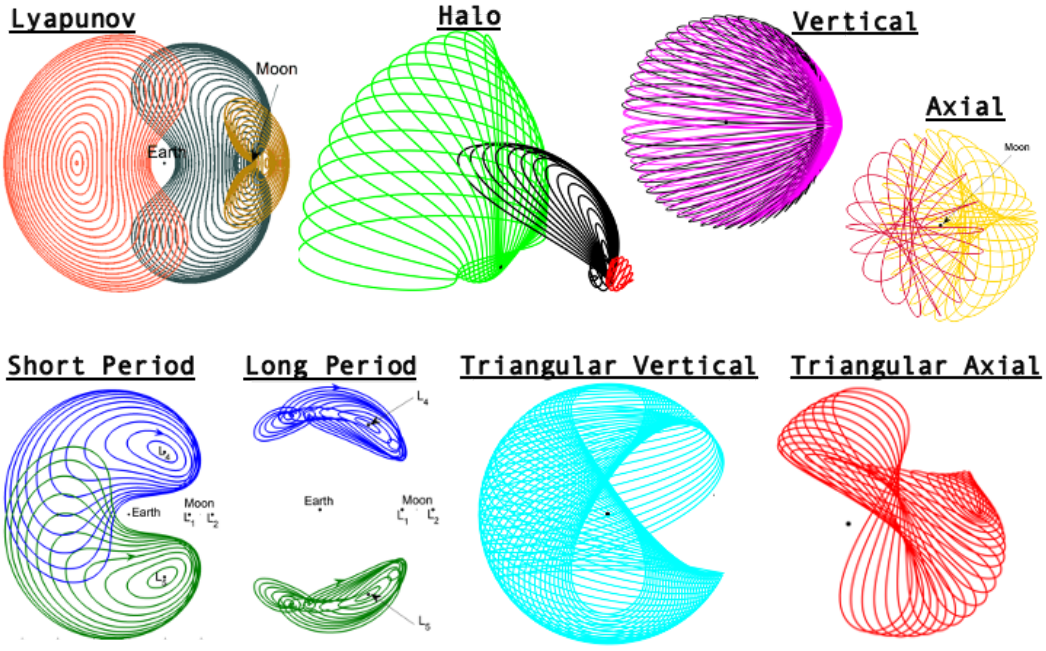


Figure 2: Examples of Periodic orbit families [1]

Some of these orbits can be advantageous for certain missions. For example the James Webb Space Telescope (JWST) is currently in a halo orbit around the Earth-Sun L2 [4]. This allows it to keep both the Earth and the Sun consistently on the same side of the craft and allows it to be far from either, both of which are very beneficial for its heat sensitive instruments. A similar orbit in the Earth-Moon system has been proposed for Habitation and Logistics Outpost (HALO) for the Lunar Gateway mission [5]. This orbit would allow the craft to maintain contact with Earth and the dark side of the moon and would enable it to act as a relay station. Many of these orbits though are not very stable. This system is also very chaotic, so a small change in initial conditions or a small perturbing effect can lead to a vastly different orbital path. This paper will examine what happens to an orbit that is periodic under ideal conditions in the Earth-Moon

system and add perturbing forces from the Sun in order to gain a better understanding of what one of these orbits would really look like.

Theory

The movement of a point mass in an idealized system can be calculated using the Circular Restricted 3 body problem (CR3BP) equations of motion. The CR3BP equations of motion can be seen in equation 1. These equations use position normalized by the distance between the primary and secondary bodies. Where μ represents the mass fraction of the secondary body relative to the total mass of the system. The distance to the primary body from the point mass is represented by d and the distance to the secondary body by r.

$$\ddot{x} - 2\dot{y} - x = -\frac{(1-\mu)(x+\mu)}{d^3} - \frac{\mu}{r^3}(x-1+\mu),$$
$$\ddot{y} + 2\dot{x} - y = -\frac{(1-\mu)}{d^3}y - \frac{\mu}{r^3}y,$$
$$\ddot{z} = -\frac{(1-\mu)}{d^3}z - \frac{\mu}{r^3}z,$$

$$d = \sqrt{(x+\mu)^2 + y^2 + z^2}, \qquad r = \sqrt{(x-1+\mu)^2 + y^2 + z^2}.$$

Equation 1: CR3BP Equations of Motion [1]

The x, y, and z coordinates of these equations and the initial conditions given by Three-Body Periodic Orbits page are in the synodic reference frame which is a non inertial reference frame where the primary and secondary bodies are fixed on the x axis and the origin is at the center of gravity of the system. This is why the equation of motion for the x axis is so different from the y and z equations. With these equations and the starting state vector of different periodic orbits given by the Three-Body Periodic Orbits page it is possible to plot a variety of these orbits. These initial conditions were used in conjunction with ephemeris data from NASA JPL SSD Horizon's page to find the position of the sun relative to an inertial frame centered at the moon [2]. Ephemeris data is location information of one celestial body relative to another. In this case that is the Sun relative to the moon. This data can come in several different formats, a vector table in the international celestial reference frame was used. The initial conditions from the Three-Body Periodic Orbits page can be transformed into an inertial frame using a simple relative motion formula because it is assumed that at the start time the Earth is on the negative x axis of the Moon inertial frame. Equation 2 shows the formulas used where body one is the

Moon, two is the spacecraft, three is the Earth-Moon barycenter, and omega is the rotational rate of the Moon around Earth in radians per second.

$$\vec{r}_{12} = \vec{r}_{32} + \vec{r}_{13}$$

$$\vec{v}_{12} = \vec{v}_{32} + \vec{v}_{13} + \Omega \, x \, \vec{r}_{32}$$

Equation 2: Relative motion equations used [6]

This makes the simulation far less accurate, but makes much of the math far easier. This still helps to give the user an idea of how unstable many of these orbits are. The formula used to compute the net acceleration due to gravity was calculated with the formula seen in equation 3. Where body 1 is the Moon, 2 is the space craft, and the summation adds the forces from the other bodies. In this case the Earth and Sun.

$$\ddot{\vec{r}}_{12} = G * \sum_{j=3}^{N} m_j * \left(\frac{\vec{r}_{1j} - \vec{r}_{12}}{r_{2j}^3} - \frac{\vec{r}_{1j}}{r_{1j}^3} \right) - \frac{G * \vec{r}_{12} * m_1}{r_{12}^3}$$

Equation 3: Acceleration due to gravity in a multibody system [7]

This equation was used in cooperation with Matlab's ODE 45 to generate the estimated path of the spacecraft through time. The inertial reference frame coordinates are then transformed back into the synodic frame by finding the distance between the spacecraft and the Earth-Moon barycenter. Then a simple z rotation by the Earth-Moon true anomaly at that time is done to align it correctly with the synodic frame. These coordinates in the synodic frame are then used to plot the orbits.

Methodology

Using all of these formulas and the data taken from SSD it was possible to create an interactive live script in Matlab. This live script allows the user to select an orbit family and Lagrange point or resonance if applicable. The user is then able to choose a random periodic orbit from this family, or a specific orbit with its ID number. The user is then able to see a few facts about the orbit. Figure 3 shows the data that a user can input.

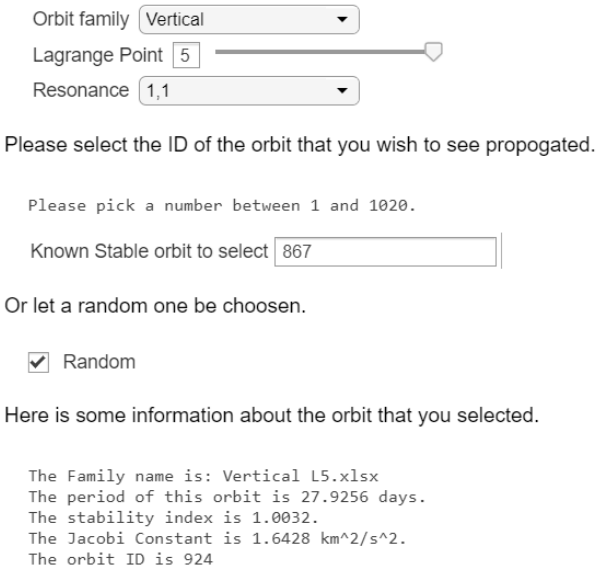


Figure 3: Available user inputs into Livescript

The user can then choose to run the program which is able to plot the orbit in the circular restricted three body problem and plot the results of the perturbations from the sun's gravity. The program plots the orbit of the spacecraft for four of the original period. This length of time was chosen because there are some orbits with very short periods of about a week to ones with a period longer than three months. Each of these orbits is then plotted in a different color to help show the flow of time. The program has one main file that the user interacts with, 3 files which do various functions, and can read from 36 spreadsheets containing most of the periodic orbits from the SSD group. Figure 4 shows a diagram of how the program is structured and a brief description of what each file does or contains.

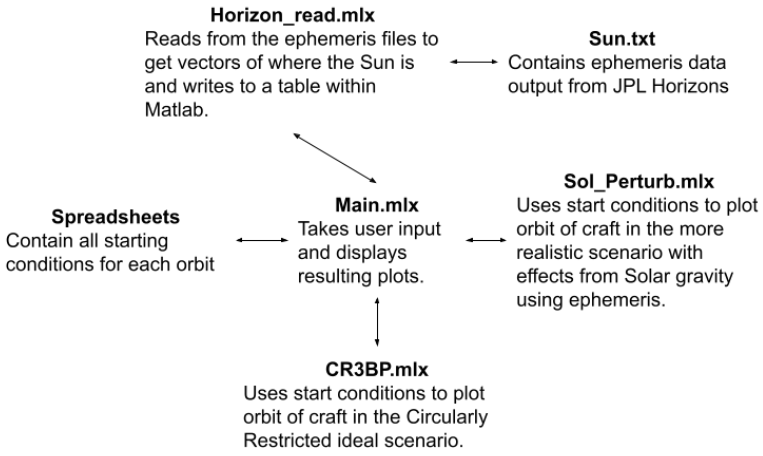


Figure 4: Diagram of how the program is structured with descriptions of each file.

The program works very well for the most part, but due to the rounding errors of Matlab and how chaotic this system is some of the results can be incorrect. For example Northern Halo L3 orbit ID 224 passes within 10 km of the Earth's center of gravity and has an orbital path that does not line up well with what SSD plots. The results of plotting this orbit can be seen in figure 5.

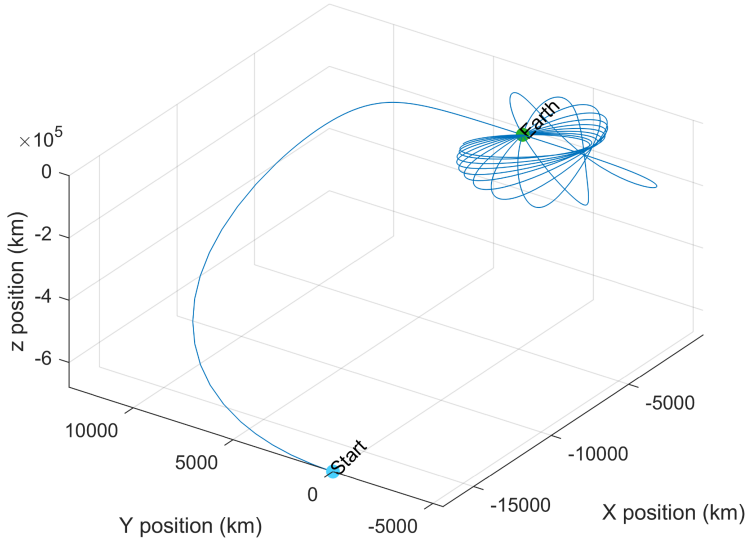


Figure 5: Plot of Northern Halo L3 orbit ID 224 without solar perturbations.

This is likely because the orbit passes so close to the Earth's center any error in the position of the spacecraft up to that point leads to a very large error in the future because the acceleration changes so much as you get slightly closer to the Earth's center. This happens with a few other orbits, but the vast majority of the orbits are well behaved without the Solar gravity perturbations. A zip file containing all of the files necessary to run the program is located here. It can be downloaded, unzipped, then the file Main.mlx can be opened and run.

Results and Discussion

Many of the orbits help to show just how chaotic the system is. Generally after 4 periods the orbit will have significantly drifted or started to follow a path that doesn't resemble the starting path much at all. For example Vertical orbit 730 around L1 has a stability index of 159.3 which is very high and so is very unstable. As figure 6 shows how the orbit starts to process around the Earth.

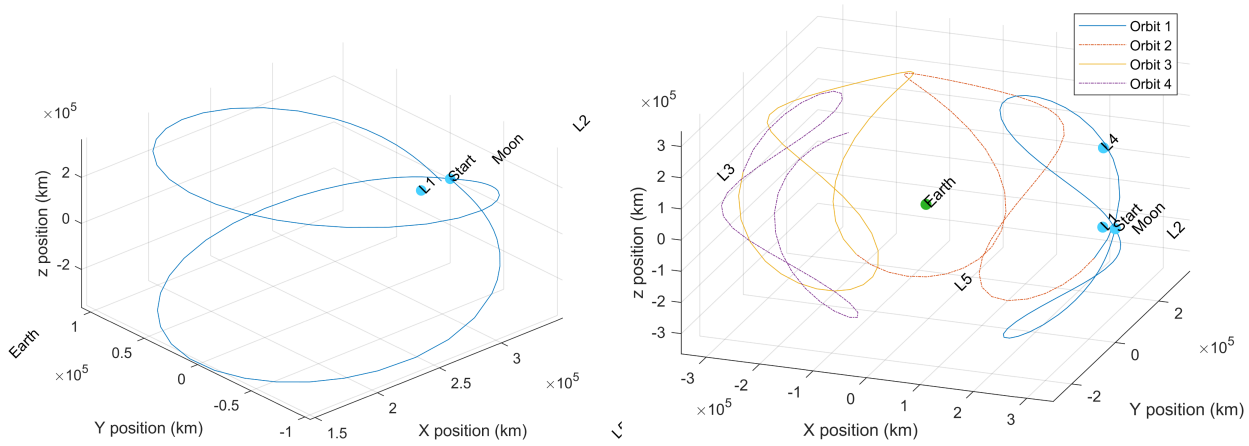


Figure 6: Vertical orbit 730 around L1. On the left is the Ideal Earth-Moon system orbit. On the right is with the addition of solar gravity perturbations.

The perturbation effects aren't nearly as evident on all of the orbits. Some of the orbits with the stability index closer to one result in more of the predicted orbit. For example Vertical orbit 187 around L3 has a stability index of 1, so is rather stable. Figure 7 shows the plots of this orbit. After 4 orbits it has started to drift pretty significantly, but still maintains a similar shape to the ideal orbit.

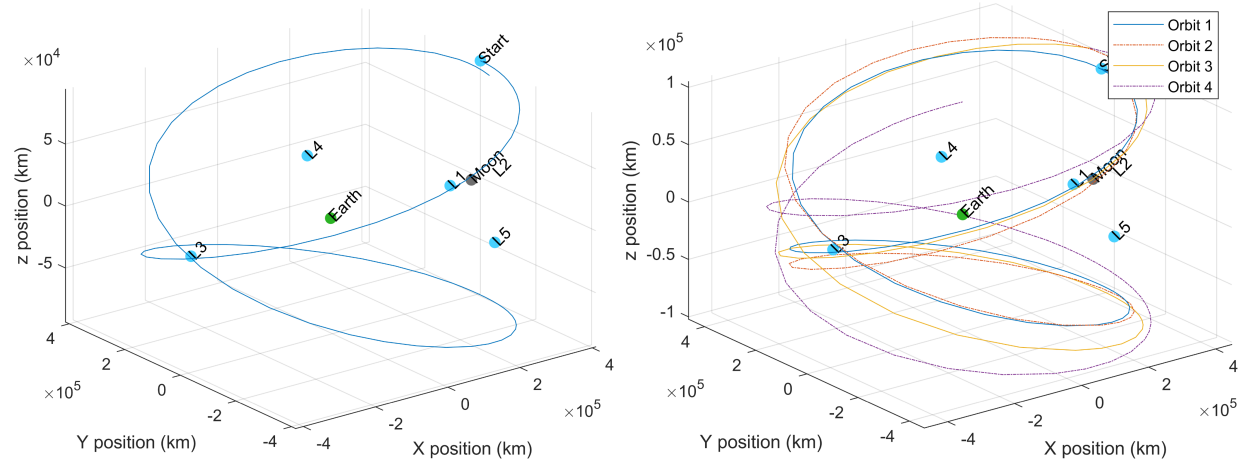


Figure 7: Vertical orbit 187 around L3. On the left is the Ideal Earth-Moon system orbit. On the right is with the addition of solar gravity perturbations.

The appendix includes several more examples of orbital plots. Though this program may not have the most accurate results, due to not using the ephemeris for all three massive bodies in the system, it still provides a good qualitative understanding of how unstable the orbits are and gives an idea of how they might be perturbed.

Future work

Given more time it would be possible to also retrieve the ephemeris data for the Earth-Moon barycenter and the Earth and use that data to propagate the orbit. This makes the coordinate transformations to and from the inertial frame much more complicated, but it would result in a flight path much closer to what would be expected to be seen if a point mass really started at that location and time. This would help reduce error because the position of the Earth and Moon relative to one another are no longer assumed, but actually measured or calculated. The Moon's orbit around Earth is nearly circular, but it still has an eccentricity of 0.0549 [8] and an inclination of about five degrees [9]. It could also be beneficial to look into using a different ODE solver to get results of the unperturbed orbits to look more similar to those generated by SSD. Overall this program turned out very well, but could use more work to get orbits more similar to what might actually be seen in a real scenario.

Conclusion

This program demonstrates very well the extent to which the orbit of a craft in a multibody system is unstable. If a craft were to use some of these orbits it would be very important to have very accurate orbit determination and course correction to ensure that the craft wouldn't start to enter an undesired orbit. Much higher accuracy simulations would need to be done if one wanted to put a spacecraft into one of these orbits, but this program would be a good starting point to build off of and provides a qualitative idea of how unstable these orbits are.

Resources

[1] Three-Body Periodic Orbits. NASA JPL, https://ssd.jpl.nasa.gov/tools/periodic_orbits.html#/intro. Accessed 17 Apr. 2023.

[2] Horizons System. NASA JPL, https://ssd.jpl.nasa.gov/horizons/app.html#/. Accessed 17 Apr. 2023.

[3] WMAP Observatory: Lagrange Points. https://map.gsfc.nasa.gov/mission/observatory_l2.html. Accessed 17 Apr. 2023.

[4] JWST Orbit - JWST User Documentation. https://jwst-docs.stsci.edu/jwst-observatory-characteristics/jwst-orbit. Accessed 17 Apr. 2023.

[5] Potter, Sean. "NASA, Northrop Grumman Finalize Moon Outpost Living Quarters Contract." NASA, 8 July 2021, http://www.nasa.gov/press-release/nasa-northrop-grumman-finalize-moon-outpost-living-quarters-contract.

[6] Curtis, Howard D. Orbital Mechanics for Engineering Students. Fourth edition, Butterworth-Heinemann, 2020.

[7] Schaub, Hanspeter, and John L. Junkins. Analytical Mechanics of Space Systems. Fourth edition, American Institute of Aeronautics and Astronautics, Inc, 2018.

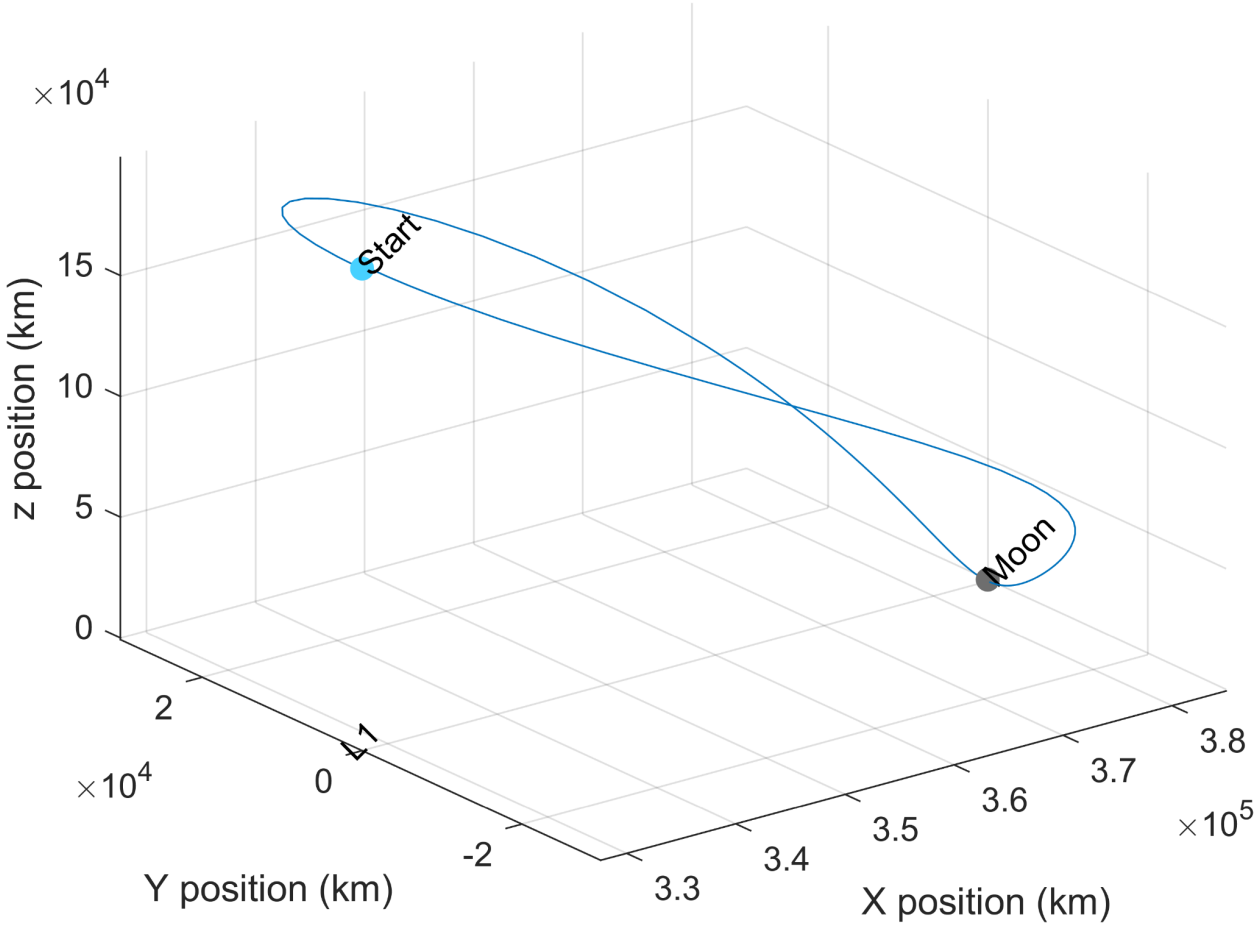
[8] Wieczorek, M. A. "The Constitution and Structure of the Lunar Interior." Reviews in Mineralogy and Geochemistry, vol. 60, no. 1, Jan. 2006, pp. 221–364. DOI.org (Crossref), https://doi.org/10.2138/rmg.2006.60.3.

[9] Lang, Kenneth R. The Cambridge Guide to the Solar System. 2nd ed, Cambridge University Press, 2011.

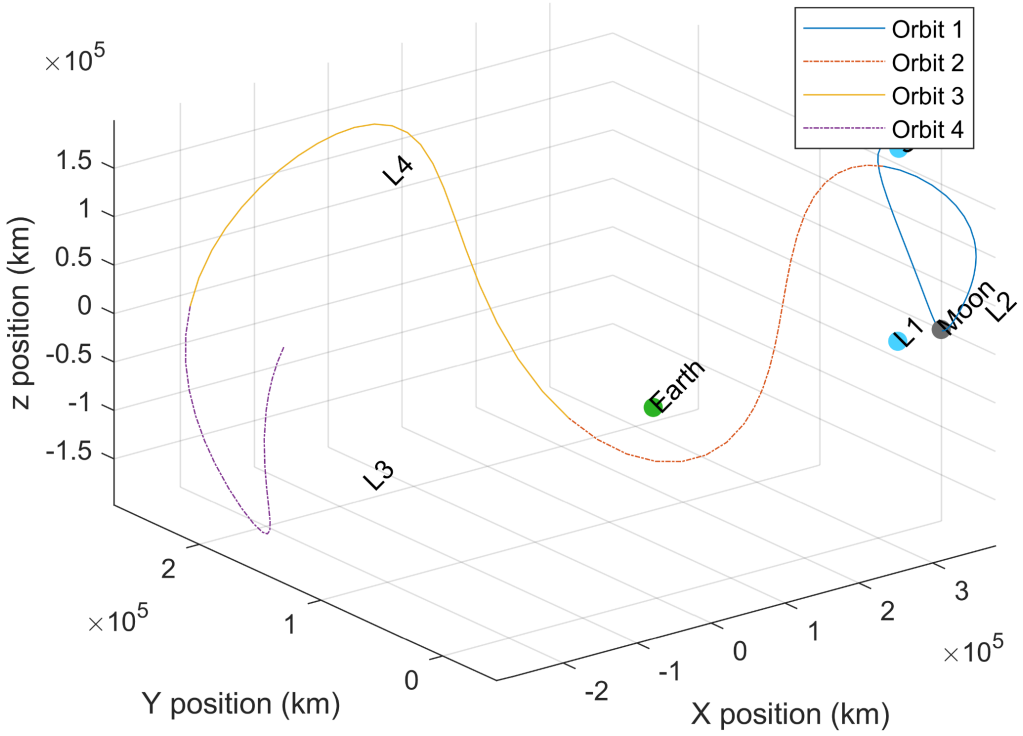
Appendix

Graphs

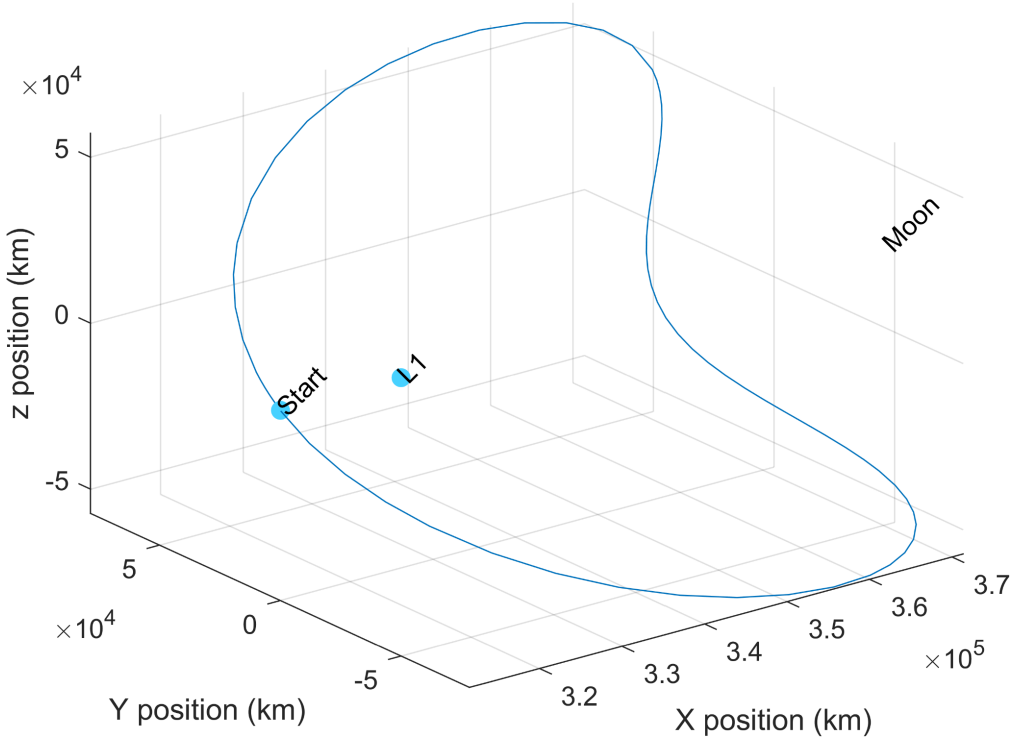
Ideal Orbit of North Halo L1 752. With a stability index of 33.6.



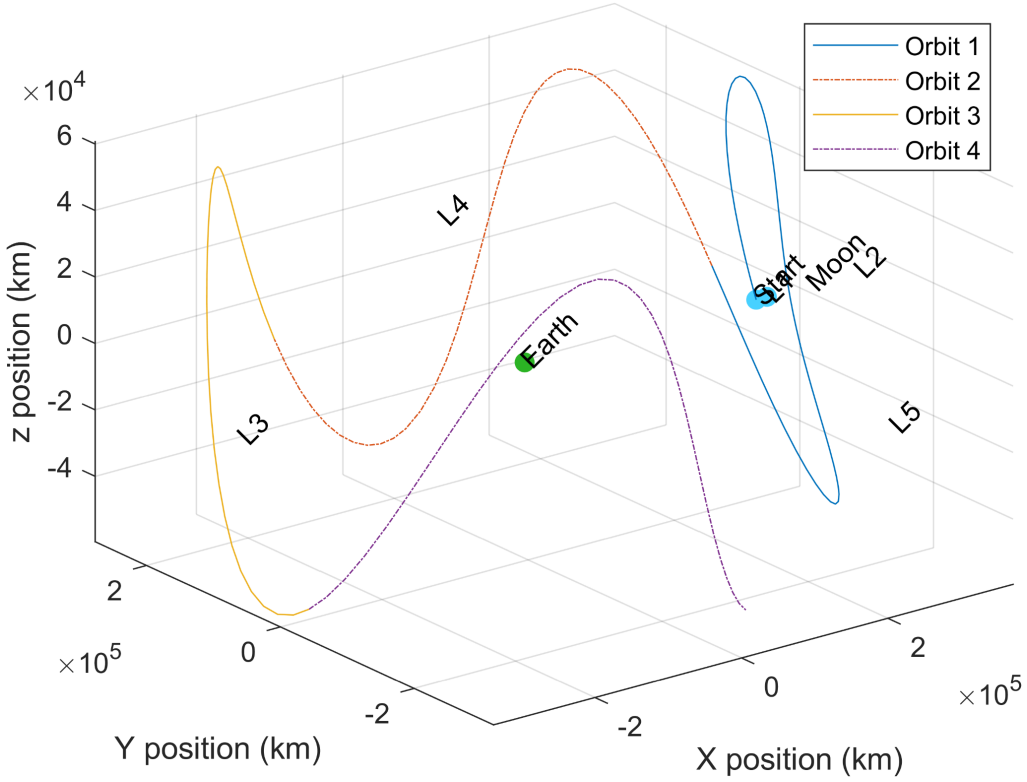
Unperturbed Orbit of North Halo L1 752



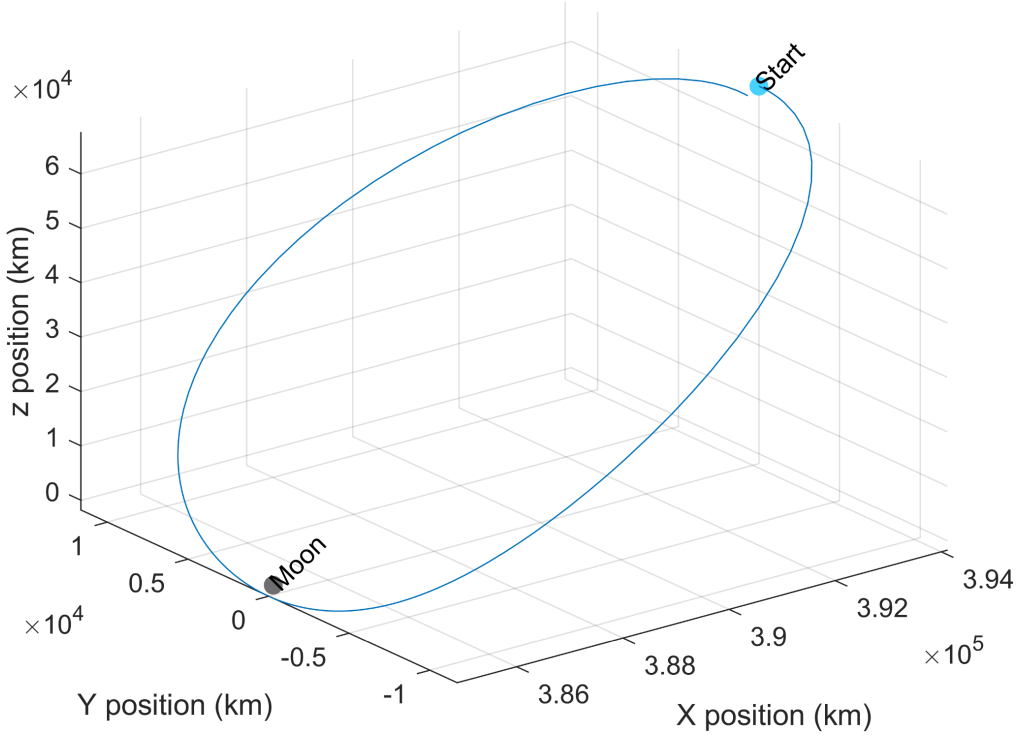
Ideal Orbit of axial L1 883. With a stability index of 220.



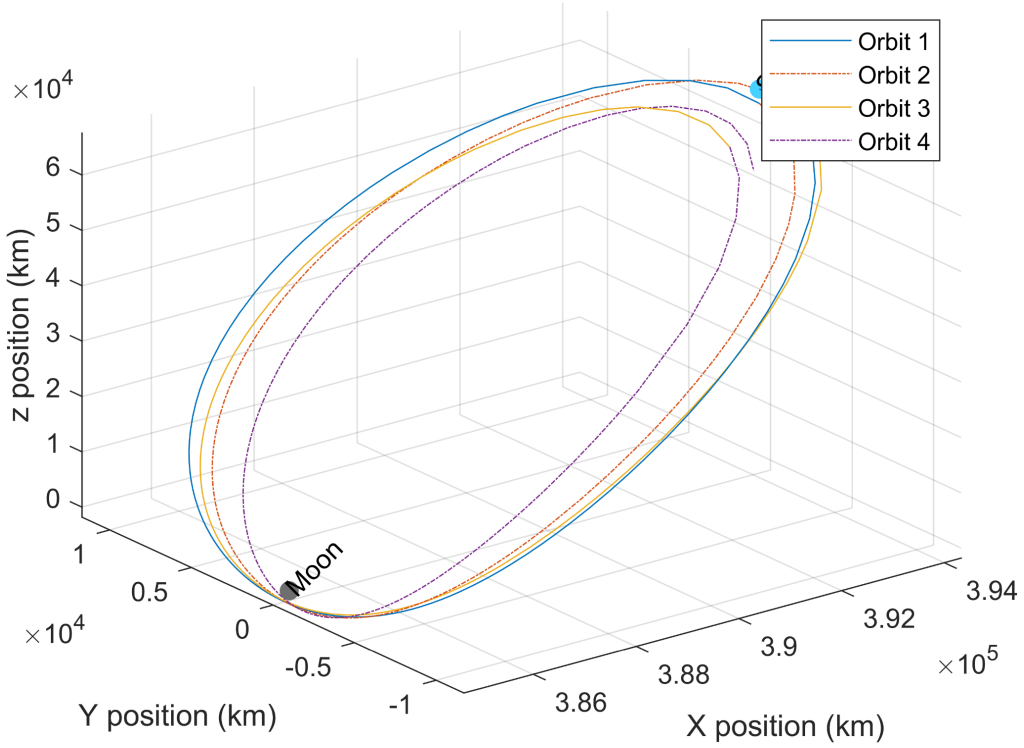
Perturbed orbit of axial L1 883



Ideal orbit of Northern Halo L2 748 with a stability index of 1.



Perturbed orbit of Northern Halo L2 748.



Matlab code

Below is the text version of each of the matlab scripts created for this program. The full project folder can be downloaded here as a .zip file here. Each of these files won't work without the correctly formatted spreadsheets to go with it.

https://drive.google.com/file/d/1ChgjhIBcg5p4lKLYzxeW68PQb2Yx87zz/view?usp=sharing

Main.mlx

Project

Cory Puckett

Select Orbit

All of the halo orbits from NASA Jet Propulsion Lab (JPL)'s Solar System Dynamics (SSD) Three body periodic halo orbits are available to choose from.

Please select an orbit family and Lagrange Point or Resonace is applicable. The families available are:

- L1, L2, L3 Lyapunov orbits

- L1, L2, L3 northern Halo orbits
- L1, L2, L3, L4, L5 Vertical orbits
- L1, L2, L3, L4, L5 Axial orbits
- L4, L5 Long Period orbits
- L4, L5 Short Period orbits
- Northern Butterfly orbits
- Northern Dragonfly orbits
- Distant Retrograde orbits
- Distant Prograde orbits
- Eastern Low Prograde orbits
- Resonant orbits

```
clearvars;

%Reads each of the ephemerides files

Horizon_read;

clearvars Lp res;

familys = ["Lyapunov", "Northern Halo", "Vertical", "Axial", "Long",
"Short", "Northern Butterfly", ...

   "Northern Dragonfly", "Distant Retrograde", "Distant Prograde",
"Eastern Low Prograde", "Resonant"];

family = familys(2);

if any(family == familys(1:2))

   lmin = 1;

   lmax = 3;

elseif any(family == familys(3:4))

   lmin = 1;

   lmax = 5;

elseif any(family == familys(5:6))

   lmin = 4;

   lmax = 5;

end

%Setting Lagrange point
```

```
if any(family == familys(1:6))
   Lp = 2; %This is the lagrange point that the user selects
else
   Lp = 0;
end
%Setting resonance
ratios = ["1,1","1,2","1,3","1,4", ...
   "2,1","2,3", ...
   "3,1","3,2","3,4", ...
   "4,1","4,3"];
if family == familys(12)
   res = ratios(1);
else
   res = "";
end
%Displaying Quantities about each orbit
% Values in table given in the following units
% Id  x0 (LU)     y0 (LU)     z0 (LU)     vx0 (LU/TU)       vy0 (LU/TU)
vz0 (LU/TU)
% Jacobi constant (LU2/TU2)  Period (TU)      Period (days)    Stability
index      Mass ratio
if Lp ~= 0
   filename = family + " L" + Lp + ".xlsx";
elseif res ~= ""
   filename = family + " " + res + ".xlsx";
else
   filename = family + ".xlsx";
end
orbits = readtable(filename);
```

Please select the ID of the orbit that you wish to see propogated.

```
%Checking if orbit num will work for each lagrange point
disp("Please pick a number between 1 and " + size(orbits,1) + ".");
```

```
Please pick a number between 1 and 1535.
```

```
orbitid = round(748); %This is the orbit ID that the user is selecting
```

Or let a random one be choosen.

```
rando = false;
if rando
   orbitid = round(rand*size(orbits,1));
end
%Constants to use
%These are used by SSD to normalize several of the values
LU = 389703; %Length Unit
TU = 382981; %Time unit
%Finds table row for selected orbit
orbit = orbits(orbitid,:);
```

Here is some information about the orbit that you selected.

```
outstring = "The Family name is: " + filename + " \n" + ...
   "The period of this orbit is " + orbit.Period_days + " days. \n" + ...
   "The stability index is " + orbit.StabilityIndex + ". \n" + ...
   "The Jacobi Constant is " + (orbit.JacobiConstant*(LU^2/TU^2)) + "
km^2/s^2. \n" + ...
   "The orbit ID is " + orbitid;
fprintf(outstring);
```

```
The Family name is: Northern Halo L2.xlsx
The period of this orbit is 6.0607 days.
The stability index is 1.
The Jacobi Constant is 3.167 km^2/s^2.
The orbit ID is 748
```

Now please choose a start date for the propogation of the orbit. Please pick a date between January 1 2000 and December 31 2029. September 15 2027 should result in some of the more

realistic results because that is when the Sun is closest to the X axis and it is a full moon, so the Sun, Earth, and Moon are all really near the X axis at the start as is assumed in the solar perturbations.

```
syear = 2027;
if syear < 2000 || syear > 2030
   disp("Please choose a year between 2000 and 2030");
end
smonth = 9;
if smonth < 1 || smonth > 12
   disp("Please choose a valid month number");
end
sday = 15;
if sday < 1 || sday > 31
   disp("Please choose a valid day number");
end
global sdate;
sdate = datetime(syear,smonth,sday); %Start date
edate = sdate + duration(24*orbit.Period_days*4,0,0); %End date
```

The unperturbed orbit.

```
CR3BP;
figure;
```

Orbit with the addition of Solar gravitational effects.

```
Sol_Perturb;
```

Sol_perturb.mlx

Solar Perturbation

Cory Puckett

This file propogades the selected orbit with the additional influence of the sun. It uses the information from the Horizon Ephemerides app. The link is below.

https://ssd.jpl.nasa.gov/horizons/app.html#/

```
clearvars -except LU TU orbit sdate edate orbitid earth sun EMBC

%Start vector relative to EMBC

x0vEMBC = [orbit.x0*LU, orbit.y0*LU, orbit.z0*LU, orbit.vx0*LU/TU,
orbit.vy0*LU/TU, orbit.vz0*LU/TU];

%Stating the mass ratio of the moon to earth

pi2 = 1.215058560962404E-2; %Mass ratio of moon

pi1 = 1 - pi2; %Mass ratio of earth

%Omega is the rotation vector of synodic reference frame

omega = [0,0,TU^-1];

%Position of Earth moon bary center

rEMBC0v = [-pi1*LU, 0,0];

vEMBC0v = cross(omega, rEMBC0v);

%Position of Earth

re0v = [-LU, 0, 0];

ve0v = cross(omega, re0v);

%This creates the state vector of the point mass relative to the moon.

x0vm = zeros([1,6]);

x0vm(1:3) = x0vEMBC(1:3) + rEMBC0v;

x0vm(4:6) = x0vEMBC(4:6) + vEMBC0v + cross(omega, x0vEMBC(1:3));
```

Do I need to add aceleration somehow?

```
%This runs the ODE45 integrator to find the state vectors through time of

%the orbit

%xmm stands for state vector matrix relative to moon

tsec = seconds(edate-sdate);

tspan = [0,tsec];

[t,xmm] = ode45(@EOM,tspan,x0vm);

%Moving xmm to be in synodic frame

xm = zeros(size(xmm,1),3);

for i = 1:size(xmm,1)
```

```
   %Finds the true anomaly and x, y, z cordinates for barycenter in moon
   %centered inertial cordinates
   theta = t(i)*TU^-1;
   xbc = -cos(theta)*LU*pi1;
   ybc = -sin(theta)*LU*pi1;
   zbc = 0;
   %radius vector
   rbc = [xbc, ybc, zbc];
%     xrot = xmm(i,1:3) * rotz(rad2deg(theta));
   xm(i,:) = (xmm(i,1:3)-rbc)*rotz(rad2deg(theta));
end
%Plotting in inertial frame for bug fixing
% figure;
% hold on;
% plot3(xmm(:,1), xmm(:,2), xmm(:,3)); %This plots the path
% scatter3(0,0,0);
% grid on;
% view(3);
%Plotting in synodic frame
figure;
hold on;
[~,iswitch,~] = unique(floor(t/(orbit.Period_days*24*60^2-1)));
syle = ["-","-.","-","-."];
for i = 1:length(iswitch)-1
   plot3(xm(iswitch(i):iswitch(i+1),1), ...
       xm(iswitch(i):iswitch(i+1),2), ...
       xm(iswitch(i):iswitch(i+1),3),syle(i));
end
grid on;
```

```
%Sets the limits of the graph
xlim([min(xm(:,1)), max(xm(:,1))]);
ylim([min(xm(:,2)), max(xm(:,2))]);
if min(xm(:,3)) == max(xm(:,3))
   zlim([-1,1]);
else
   zlim([min(xm(:,3)), max(xm(:,3))]);
end
%Using x and y locations for lagrange points from SSD
xeq = [0.83691513, 1.15568217, -1.00506265, 0.48784941, 0.48784941]*LU;
yeq = [0,0,0, 0.86602540, -0.86602540]*LU;
%Finding earth and moon position
r12 = LU; %Distance between Moon and the Earth in km
x1 = -pi2*r12;
x2 = pi1*r12;
%Defining extra point location to plot
xpoints = horzcat(x1,x2,xeq, orbit.x0*LU);
ypoints = horzcat(0,0,yeq, orbit.y0*LU);
zpoints = horzcat(zeros(1,7), orbit.z0*LU);
%Plotting each point
hold on;
c = vertcat([43, 179, 36; 110, 110, 110], ones(6,3).*[71, 209, 255]);
%Data point colors
c = c/255;
scatter3(xpoints,ypoints,zpoints,50,c,"filled");
%Labeling each point
dx = 0;
dy = 0;
words = ["Earth","Moon","L1","L2","L3","L4","L5", "Start"];
labels = text(xpoints+dx,ypoints+dy,zpoints,words);
```

```
set(labels,'Rotation',45);
%Labeling each axis
xlabel("X position (km)");
ylabel("Y position (km)");
zlabel("z position (km)");
view(3);
%Adding legend
legend("Orbit 1", "Orbit 2", "Orbit 3", "Orbit 4","Location","northeast");
function dxdt = EOM(t,X)
   %Adding in needed global variables
   global sun sdate

   %Constants
   me = 5.97219*10^24; %Mass of the Earth
   ms = 1988500*10^24; %Mass of the Sun
   mm = 7.349*10^22; %Mass of the moon
   G = 6.67430*10^-20; %Newtonian constant of gravitation in km^3/(kg*s^2)
from
   %https://ssd.jpl.nasa.gov/astro_par.html
   TU = 382981; %Time unit
   LU = 389703; %Length Unit

   %Current time
   time = sdate + duration(0,0,t);

   %Method to use for interpolation of points
   method = 'pchip';

   %Creating state values of earth
   %Earth starts at -LU because that's how it's set up in the synodic
```

```
%frame.
theta = t*TU^-1;
xe = -cos(theta)*LU;
ye = -sin(theta)*LU;
ze = 0;

%Vector to earth following formula naming convention
r13 = [xe, ye, ze];

%Creating position values of sun
xs = interp1(sun.Date,sun.X,time,method);
ys = interp1(sun.Date,sun.Y,time,method);
zs = interp1(sun.Date,sun.Z,time,method);

%Vector to sun following formula naming convention with sun number 4
r14 = [xs,ys,zs];

%Giving variables more easily readable names
x = X(1);
y = X(2);
z = X(3);
xdot = X(4);
ydot = X(5);
zdot = X(6);

%r vector follows formula naming convention
r12 = [x,y,z];

%Defining output vector
```

```
    dxdt = zeros(6,1);


    %Giving values to output vector
    dxdt(1) = xdot;
    dxdt(2) = ydot;
    dxdt(3) = zdot;


    %Formula with only moon influence
%       dxdt(4:6) = -G*mm*r12/(norm(r12)^3);


    %Formula with only moon and earth influence
%       r23 = -r13 + r12;
%       dxdt(4:6) = G*me*((r13-r12)/(norm(r23)^3) - r13/(norm(r13))^3) -
G*mm*r12/(norm(r12)^3);


    %Formula with only moon and earth influence
    r23 = -r13 + r12;
    r24 = -r14 + r12;
    %Formula for accelaration from moon, earth, and sun
    am = -G*mm*r12/(norm(r12)^3);
    ae = G*me*((r13-r12)/(norm(r23)^3) - r13/(norm(r13))^3);
    as = G*ms*((r14-r12)/(norm(r24)^3) - r14/(norm(r14))^3);
    dxdt(4:6) = as + ae + am;
end
```

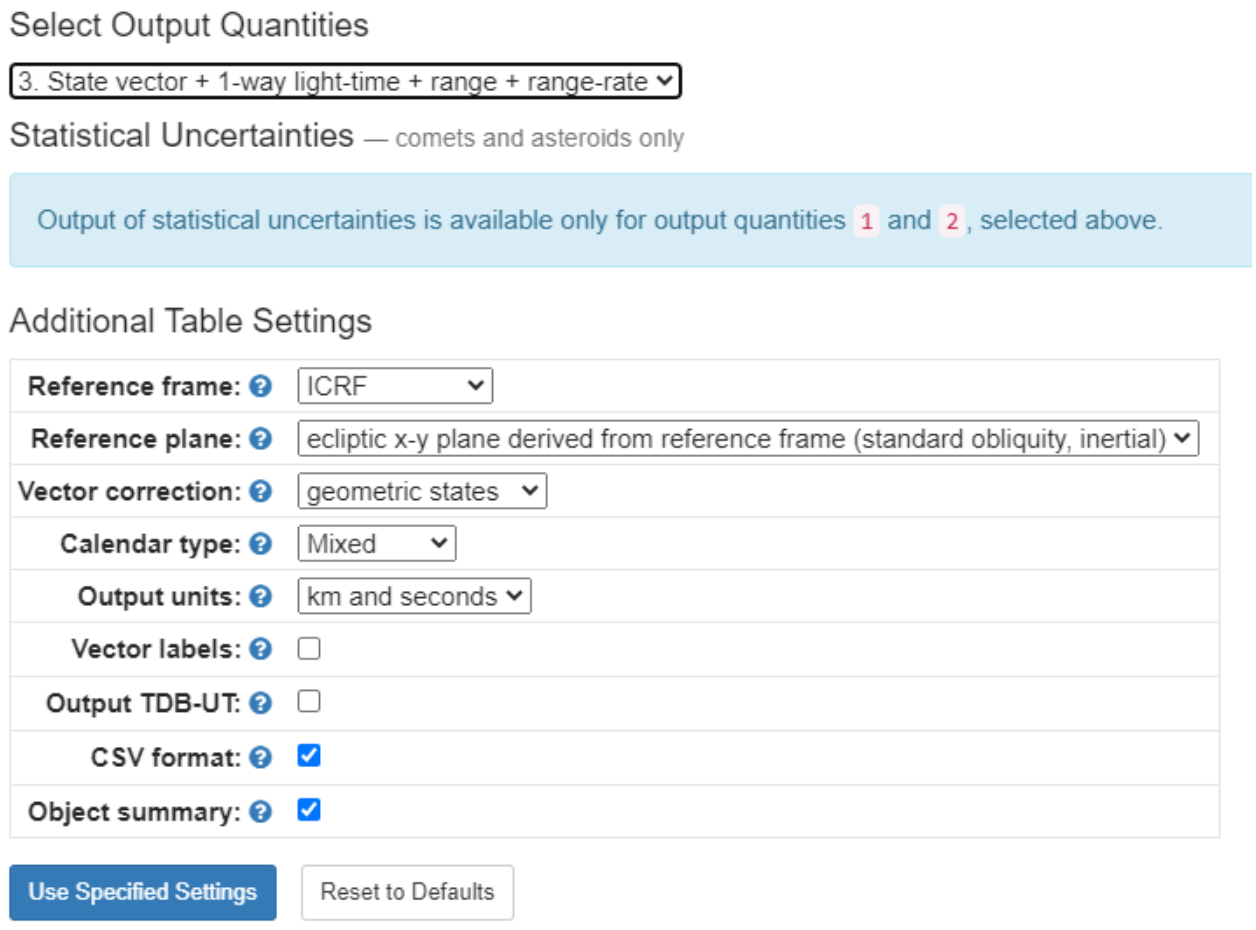
Horizon_read.mlx

Read Horizon files

Cory Puckett

This file will take the output of Horizons and reads each file to a table that can be read later in the program.

When generating the files in horizons use the Moon (body center) as the cordinate center and use the below table settings. In order for everthing to work correctly you need to choose a start date in AD in Horizons.



Earth.txt contains the position vector data of earth relative to the moon from Jan-1-2000 to Dec-31-2030.

Sun.txt contains the position vector data of the Sun relative to the moon from Jan-1-2000 to Dec-31-2030.

EMBC.txt contains the position vector data of the Earth-Moon Bary center (EMBC) relative to the moon from Jan-1-2000 to Dec-31-2030.

```
clear opts;

opts = delimitedTextImportOptions('NumVariables',11);
```

```
%This sets up how many columns there are
opts.Delimiter = [",",", A.D."];
%This sets up the delimiters. The AD is there to remove it because I
%couldn't find a format to read it
opts.VariableNames = ["JDTDB", "Date", "X", "Y", "Z", "VX", "VY", "VZ",
"LT", "RG", "RR"];
%Colmun names
opts = setvartype(opts,{'double'});
%Sets all data types to doubles
opts = setvartype(opts,"Date",{'datetime'});
%Changes Date to datetime type
opts = setvaropts(opts,"Date","InputFormat","yyy-MMM-dd HH:mm:SS.ssss");
%Sets up input format for date and time of each position vector
%Declaring variables as global, so that they can be used in functions
global earth sun EMBC
%This then reads each of the files and saves it by the name and at the
%needed start point
for i = 3 %Change this to 1:3 if you want to read all three files.
   if i == 1
       opts.DataLines = 55;
       earth = readtable("Earth.txt",opts);
       earth = earth(isfinite(earth.JDTDB),1:11);
       %This line trims the lines talking about cordinates at the end of
each file.
   elseif i == 2
       opts.DataLines = 35;
       EMBC = readtable("EMBC.txt",opts);
       EMBC = EMBC(isfinite(EMBC.JDTDB),1:11);
       %EMBC is Earth Moon Bary Center
   else
```

```
        opts.DataLines = 55;
        sun = readtable("Sun.txt",opts);
        sun = sun(isfinite(sun.JDTDB),1:11);
    end
end
```

CR3BP.mlx

Circlular Restricted 3 Body Problem Propogation

Cory Puckett

This script will take the given starting parameters of the orbit and plot them in the main file. This file uses the same normalized units as SSD.

The vector below is used to go into the cr3bp EOM to propogate and find the orbit of it.

```
x0v = [orbit.x0, orbit.y0, orbit.z0, orbit.vx0, orbit.vy0, orbit.vz0];
```

This vector contains the information $[r_x, r_y, r_z, v_x, v_y, v_z]$ for the start point.

```
tf = orbit.Period; %Change in time from first to last point
%This is based on the period in days given by SSD
```

This section will throw an error if the spreadsheet got save incorrectly and each of the numbers only has 3 digits.

```
digits = zeros(size(x0v));
for i = 1:length(x0v)
    digits(i) = numel(num2str(x0v(i)));
end
if digits <= 4
    error("Check current spreadsheet it may have been saved incorrectly
and" + ...
        "may not have enough decimal places");
end
```

xm is the matrix containing all of the x locations through time.

xmn contains all of the normalized position and velocity data

```
[t,xmn] = ode45(@cr3bp,[0,tf],x0v);
xm = [xmn(:,1:3)*LU, xmn(:,4:6)*LU/TU];
plot3(xm(:,1), xm(:,2), xm(:,3)); %This plots the path
grid on;
%Sets the limits of the graph
xlim([min(xm(:,1)), max(xm(:,1))]);
ylim([min(xm(:,2)), max(xm(:,2))]);
if min(xm(:,3)) == max(xm(:,3))
   zlim([-1,1]);
else
   zlim([min(xm(:,3)), max(xm(:,3))]);
end
%Using x and y locations for lagrange points from SSD
xeq = [0.83691513, 1.15568217, -1.00506265, 0.48784941, 0.48784941]*LU;
yeq = [0,0,0, 0.86602540, -0.86602540]*LU;
%Finding earth and moon position
r12 = 389703; %Distance between Moon and the Earth in km
pi2 = 1.215058560962404E-2;
pi1 = 1 - pi2;
x1 = -pi2*r12;
x2 = pi1*r12;
%Defining extra point location to plot
xpoints = horzcat(x1,x2,xeq, orbit.x0*LU);
ypoints = horzcat(0,0,yeq, orbit.y0*LU);
zpoints = horzcat(zeros(1,7), orbit.z0*LU);
%Plotting each point
hold on;
c = vertcat([43, 179, 36; 110, 110, 110], ones(6,3).*[71, 209, 255]);
%Data point colors
```

```
c = c/255;
scatter3(xpoints,ypoints,zpoints,50,c,"filled");
%Labeling each point
dx = 0;
dy = 0;
words = ["Earth","Moon","L1","L2","L3","L4","L5", "Start"];
labels = text(xpoints+dx,ypoints+dy,zpoints,words);
set(labels,'Rotation',45);
%Labeling each axis
xlabel("X position (km)");
ylabel("Y position (km)");
zlabel("z position (km)");
```

This function contains the equation of motion for the circular restricted 3 body problem for the Earth Moon System.

```
function dxdt = cr3bp(~,X)
   %Mass ratio
   mu = 1.215058560962404E-2;

   %Giving variables more easily readable names
   x = X(1);
   y = X(2);
   z = X(3);
   xdot = X(4);
   ydot = X(5);
   zdot = X(6);

   %Equations for d and r
   d = sqrt((x+mu)^2 + y^2 + z^2); %Distance from primary body
   r = sqrt((x-1+mu)^2 + y^2 + z^2); %Distance from secondary body
```

```
    %Defining output vector
    dxdt = zeros(6,1);
```

These formulas are from SSD introduction page

https://ssd.jpl.nasa.gov/tools/periodic_orbits.html#/intro

```
    dxdt(1) = xdot;
    dxdt(2) = ydot;
    dxdt(3) = zdot;
    dxdt(4) = -(1-mu)*(x+mu)/d^3 - (mu/r^3)*(x-1+mu) + 2*ydot + x;
    %Formula for x acceleration
    dxdt(5) = -(1-mu)*y/d^3 - (mu/r^3)*y - 2*xdot + y;
    %Formula for y acceleration
    dxdt(6) = -(1-mu)*z/d^3 - (mu/r^3)*z;
    %Formula for z acceleration
end
```