Table of Contents


```
%Roshan Jaiswal-Ferri, Stefan Rosu, Michiru Warren, Benjamin Howard
%Section - 01
%Aero 351: Space Debris Removal - 11/13/24
```

Workspace Prep

```
format long      %Allows for more accurate decimals
close all;       %Clears all
clear all;       %Clears Workspace
clc;             %Clears Command Window
```

Global Vars

```
mu = 398600;
Rearth = 6378;
tol = 1e-8;
```

Reading TLE Data

```
%Satellite Selection:
%Leo 1: COURIER 1B
```

```
%Leo 2: COURIER 1B ROCKET
%Meo: H2SAT (HEINRICH HERTZ)
%Geo: GALAXY 14 (G-14)


% Read TLE data using the new readTLE function
tleL1 = readTLE("TLE_Data\LEO1.txt");
tleL2 = readTLE("TLE_Data\LEO2.txt");
tleM = readTLE("TLE_Data\MEO.txt");
tleG = readTLE("TLE_Data\GEO.txt");

% Extract inclination
incL1 = tleL1.inclination;
incL2 = tleL2.inclination;
incM = tleM.inclination;
incG = tleG.inclination;

% Extract eccentricity
eccL1 = tleL1.eccentricity;
eccL2 = tleL2.eccentricity;
eccM = tleM.eccentricity;
eccG = tleG.eccentricity;

% Extract Right Ascension of Ascending Node (RAAN)
RAANL1 = tleL1.rightAscension;
RAANL2 = tleL2.rightAscension;
RAANM = tleM.rightAscension;
RAANG = tleG.rightAscension;

% Extract argument of periapsis
ArgPL1 = tleL1.argumentOfPerigee;
ArgPL2 = tleL2.argumentOfPerigee;
ArgPM = tleM.argumentOfPerigee;
ArgPG = tleG.argumentOfPerigee;

%Me given in rev/day:
MeL1 = tleL1.meanAnomaly*((2*pi)/86400); %Me in rad/s
MeL2 = tleL2.meanAnomaly*((2*pi)/86400);
MeM = tleM.meanAnomaly*((2*pi)/86400);
MeG = tleG.meanAnomaly*((2*pi)/86400);

%true anomaly
nuL1 = MetoNu(MeL1,eccL1);
nuL2 = MetoNu(MeL2,eccL2);
nuM = MetoNu(MeM,eccM);
nuG = MetoNu(MeG,eccG);

%semi major axis
aL1 = Me2a(tleL1.meanMotion,mu);
aL2 = Me2a(tleL2.meanMotion,mu);
aM = Me2a(tleM.meanMotion,mu);
aG = Me2a(tleG.meanMotion,mu);

%Juilian Date
```

```
JDtimeL1 = juliandate(2024,11,18,18,01,04);
JDtimeL2 = juliandate(2024,11,17,23,04,40);
JDtimeM = juliandate(2024,11,13,02,18,12);
JDtimeG = juliandate(2024,11,13,11,24,12);
JDStart = juliandate(2024,11,22,0,0,0); %start nov-22-2024 at midnight
```

PART 3: Finding State Vectors (R & V) & Propogation

```
%INPUT AND OUTPUT IN METERS!!!!!!
[R,V] = keplerian2ijk(aL1*1000,eccL1,incL1,RAANL1,ArgPL1,nuL1);
R = R./1000;
V = V./1000;
```

Note for Dr. A:

```
%We used keplerian2ijk for help debugging and cross checking, later in the
%script you can see us use our own coes2rvd (degrees) function that we
%coded and which outputs exactly the same answer! :)

[~,~,~,~,~,~,~,p] = rv2coes(R,V,mu,Rearth);

PropTime = (JDStart-JDtimeL1)+(5*p); %seconds
timespan = [0, PropTime];
JDCurrent = PropTime;
state = [R, V];
%INPUTS MUST BE IN THAT ORDER UNTIL OPTIONS
options = odeset('RelTol',1e-8,'AbsTol',1e-8);

[~,Orbit1] = ode45(@twobodymotion,timespan,state,options,mu);

RL1 = [Orbit1(end,1),Orbit1(end,2),Orbit1(end,3)];
VL1 = [Orbit1(end,4),Orbit1(end,5),Orbit1(end,6)];
```

PART 4: First Transfer

```
[~,~,~,~,~,~,~,p,t,~,Ra] = rv2coes(RL1,VL1,mu,Rearth);

tA = (p/2)-t; %time to apogee
JDCurrent = JDCurrent + tA;

%propogation for apogee
state = [RL1, VL1];
tspan = [0, tA];
[~,Orbit1A] = ode45(@twobodymotion,tspan,state,options,mu);

RL1A = [Orbit1A(end,1),Orbit1A(end,2),Orbit1A(end,3)]; %R&V at apogee
VL1A = [Orbit1A(end,4),Orbit1A(end,5),Orbit1A(end,6)];

rT = norm(Orbit1A(end,1:3)); %R target
```

```
%Circ Burn
v = sqrt(mu/rT);
dVc = v-norm(VL1A); %delta v from circ burn

uV = VL1A/norm(VL1A); %unit vector
dVd = dVc*uV; %delta V with direction
VL1C = VL1A+dVd; %new velocity vector after adding velocity
RL1C = RL1A;

hL1C = cross(RL1C,VL1C); %h of Leo 1 circ orbit

[~,a,e,nuL12,~,~,w,~] = rv2coes(RL1A,VL1C,mu,Rearth); %e confirms we are
circ!

%INC + RAAN Change
dR = abs(RAANL1-RAANL2);
alpha = acosd(cosd(incL1)*cosd(incL2)+sind(incL1)*sind(incL2)*cosd(dR));
dVincR = 2*norm(VL1C)*sind((alpha/2)); %check VL1C if something breaks (and
are desperate)

dVt = dVincR + dVc; %delta V total


[R2,V2] = keplerian2ijk(a*1000,e,incL2,RAANL2,w,rad2deg(nuL12));
R2 = R2./1000; %random RV of new orbit with inc + RAAN
V2 = V2./1000;

hL12_2 = cross(R2,V2); %h of ht orbit

nodeVect12 = cross(hL1C,hL12_2);
nodeVect12U = nodeVect12/norm(nodeVect12); %node line unit vector
RL12B = nodeVect12U*norm(RL1C); %position of inc raan burn (node line)
V12u = cross(hL12_2,RL12B)/norm(cross(hL12_2,RL12B)); %unit vector vel
VL12B = norm(VL1C)*V12u; %velocity at position of inc raan burn (after)

%finding time of inc raan burn for correct propagation time
theta = acos((dot(RL12B,RL1A))/... %in radians!!
    (norm(RL1A)*norm(RL12B))); %Angle between apoapse & inc raan change
S = theta*norm(RL1A); %arclength
tS = S/norm(VL1C); %in seconds (Time from circ burn till node line/inc burn)
JDCurrent = JDCurrent + tS;
```

Propogation of Orbit 1 to 2

```
JDL2 = abs(JDtimeL1-JDtimeL2);
dtL = 5*p;
Ttrnsfr = PropTime+dtL+JDL2;
tspan = [0,Ttrnsfr];
tspan12 = [0, tS];
[R22,V22] = keplerian2ijk(aL2*1000,eccL2,incL2,RAANL2,ArgPL2,nuL2);
R22 = R22./1000;
V22 = V22./1000;
stateL2tle = [R22, V22];
```

```
stateL2 = [RL1A, VL1C];

%prop of circularized burn
[timeNewL2,Trans12_1] = ode45(@twobodymotion,tspan12,stateL2,options,mu);
RL2 = [Trans12_1(end,1),Trans12_1(end,2),Trans12_1(end,3)];
VL2JD = [Trans12_1(end,4),Trans12_1(end,5),Trans12_1(end,6)];

%-------------------------
```

Matching ecc Orbit 1 to 2 for HT

```
[~,~,~,~,~,~,~,pL2,t2,~,Ra2] = rv2coes(R22,V22,mu,Rearth);

tA2 = (pL2/2)-t2; %time to apogee orbit 2

state2 = [R22, V22];
tspan2 = [0, tA2];
[~,Orbit2A] = ode45(@twobodymotion,tspan2,state2,options,mu);

R2A = [Orbit2A(end,1),Orbit2A(end,2),Orbit2A(end,3)]; %R&V at apogee orb 2
V2A = [Orbit2A(end,4),Orbit2A(end,5),Orbit2A(end,6)];

[RL1CP,VL1CP] = keplerian2ijk(a*1000,e,incL2,RAANL2,ArgPL2,0);
RL1CP = RL1CP./1000;
VL1CP = VL1CP./1000; %R & V vectors of circ orbit at periapse (opp of orb 2
apogee)

theta2 = acos((dot(RL12B,RL1CP))/... %in radians!!
    (norm(RL1CP)*norm(RL12B))); %Angle between inc raan change & peri of orb
2
SHB1 = theta2*norm(RL1CP); %arclength
tSHB1 = SHB1/norm(VL1C); %in seconds (Time from inc burn to ht burn 1)
JDCurrent = JDCurrent + tSHB1;

% Propogation of second transfer orbit (inc raan change)
tspan12_2 = [0, tSHB1];
state12_2 = [RL12B, VL12B];

[timeNew12_2,Trans12_2] =
ode45(@twobodymotion,tspan12_2,state12_2,options,mu);
R12_2 = [Trans12_2(end,1),Trans12_2(end,2),Trans12_2(end,3)];
V12_2 = [Trans12_2(end,4),Trans12_2(end,5),Trans12_2(end,6)];


%Transfer Orbit Coes (HT)
ecc12 = (norm(RL1C)-norm(R2A))/(norm(RL1C)+norm(R2A));
aH1 = norm(R2A)/(1-ecc12);

[RH1_1,VH1_1] = keplerian2ijk(aH1*1000,ecc12,incL2,RAANL2,ArgPL2,0);
RH1_1 = RH1_1./1000;
VH1_1 = VH1_1./1000;

dVH1B1 = abs(norm(VH1_1) - norm(VL1CP)); %delta v from ht burn 1
```

```
dVt = dVt + dVH1B1;

[~,~,~,~,~,~,~,p3] = rv2coes(RH1_1,VH1_1,mu,Rearth);

tH = p3/2; %time of ht orbit
JDCurrent = JDCurrent + tH; %Ends with us at apogee

stateH = [RH1_1,VH1_1];
tspanH = [0,tH];
[timeNew,HT1] = ode45(@twobodymotion,tspanH,stateH,options,mu);

RH1_2 = [HT1(end,1),HT1(end,2),HT1(end,3)]; %R&V at apogee orb 2
VH1_2 = [HT1(end,4),HT1(end,5),HT1(end,6)];

dVH1B2 = abs(norm(VH1_2)-norm(V2A));
dVt = dVt + dVH1B2;

%Final vel is V2A
```

Propogate half orbit 2 of us to perigee for phase change

```
JDCurrent = JDCurrent + pL2/2; %Updating jd to move from apo to peri

tspanJD = [0, JDCurrent];
%prop of leo orbit 2 (rocket body) ending when ht arrives
[~,Orbit2] = ode45(@twobodymotion,tspanJD,stateL2tle,options,mu);
RL2JD = [Orbit2(end,1),Orbit2(end,2),Orbit2(end,3)];
VL2JD = [Orbit2(end,4),Orbit2(end,5),Orbit2(end,6)];

state2A = [R2A,V2A];
tspanUS = [0, 0.5*pL2];
%prop of us moving from apogee to perigee
[~,ORB2P] = ode45(@twobodymotion,tspanUS,state2A,options,mu);
RORB2P = [ORB2P(end,1),ORB2P(end,2),ORB2P(end,3)];
VORB2P = [ORB2P(end,4),ORB2P(end,5),ORB2P(end,6)];
```

ORBIT 2 Propogation For Rendezvous

```
[~,~,~,nuL2JD,~,~,~,pL2] = rv2coes(RL2JD,VL2JD,mu,Rearth); %finding nu of
rocket b
nuL2JDd = rad2deg(nuL2JD); %rv2coes outputs rads

[~,~,~,nuJD] = rv2coes(RORB2P,VORB2P,mu,Rearth); %finding nu of us
nuJDd = rad2deg(nuJD);
```

New Calcs for phase change

```
%Orbit 1: (Ours)
Rp1 = norm(RORB2P); %km
Ra1 = norm(R2A); %km
```

```
theta1B = nuJDd; %deg
theta1C = nuL2JDd/5; %deg
ecc1 = eccL2;
a1 = aL2;
h2 = sqrt(mu*a1*(1-ecc1^2));
T1 = pL2;

Rb1 = ((h2^2)/mu)/(1+ecc1*cosd(theta1B)); %current position of s/c B
Vb1t = (mu/h2)*(1+ecc1*cosd(theta1B)); %V tangential
Vb1r = (mu/h2)*ecc1*sind(theta1B); %V radial
Vt1 = sqrt((Vb1t^2)+(Vb1r^2));
FPAb1 = atand(Vb1r/Vb1t); %flight path angle in deg

%For us
E = 2*atan(sqrt((1-ecc1)/(1+ecc1))*tand(theta1B/2)); %eccentric anomaly
Me = E-ecc1*sin(E); %Mean anomaly
ts = (Me*T1)/(2*pi); %time since periapsis

%For Leo 2 (Courier 1B RB)
E2 = 2*atan(sqrt((1-ecc1)/(1+ecc1))*tand(theta1C/2)); %eccentric anomaly
Me2 = E2-ecc1*sin(E2); %Mean anomaly
ts2 = (Me2*T1)/(2*pi); %time since periapsis

%Orbit 2: (Phase Change Orb)
dt = T1-(ts2-ts); %period of phase change transfer orbit
a2 = (dt^(2/3)*mu^(1/3))/(2^(2/3)*pi^(2/3));
Rp2 = norm(RORB2P);
Ra2 = 2*a2-Rp2;
ecc2 = (Rp2-Ra2)/(Ra2+Rp2); %<3: is elliptical
h2 = sqrt(mu*a2*(1-ecc2^2));
Vt2 = h2/Rp2;

%DeltaV EQ:
%2nd Flight path angle is 0:
dV = 2*sqrt((Vt1^2)+(Vt2^2)-2*Vt1*Vt2*cosd(0-FPAb1));
%This is both burns combined (each burn for PC is half of this)
```

Calculating delta v and propogating phase change

```
%dV = VP - VORB2P'; %calculating the dV value
VORB2PU = VORB2P/norm(VORB2P);
dV2 = VORB2PU*(dV/2);

dVt = dVt + norm(dV); %adding dV from both burns

dt = dt*5;

tspanP = [0,dt];
stateP = [RORB2P,(VORB2P-dV2)];
[~,Phase1] = ode45(@twobodymotion,tspanP,stateP,options,mu);
```

```
RP1 = [Phase1(end,1),Phase1(end,2),Phase1(end,3)]; %R&V at end of phase
VP1 = [Phase1(end,4),Phase1(end,5),Phase1(end,6)];
```

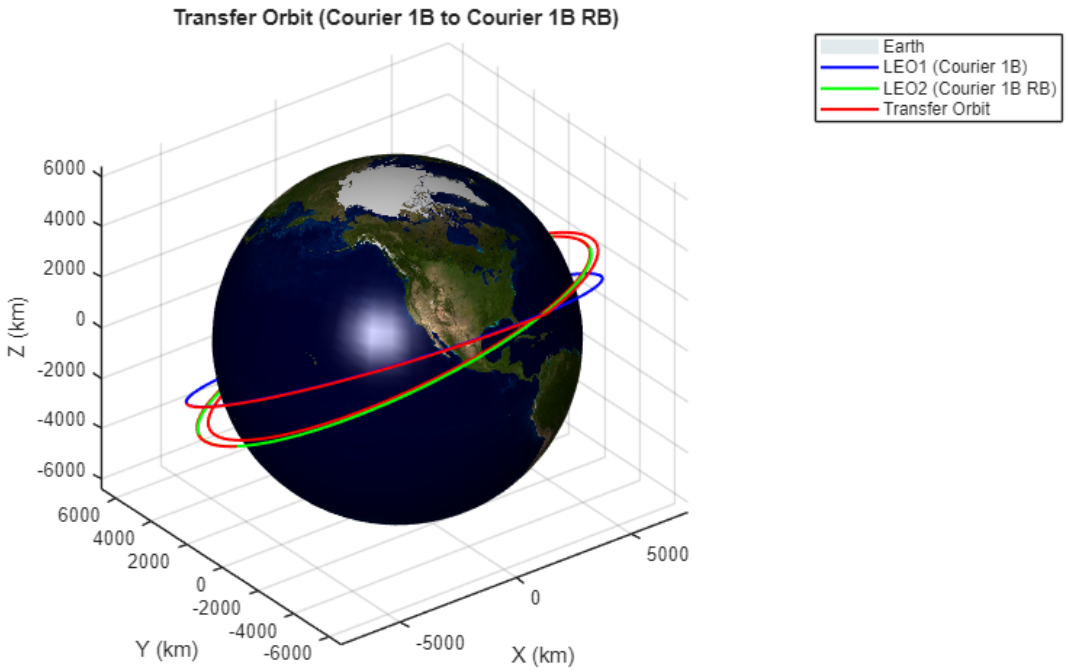
Propogation

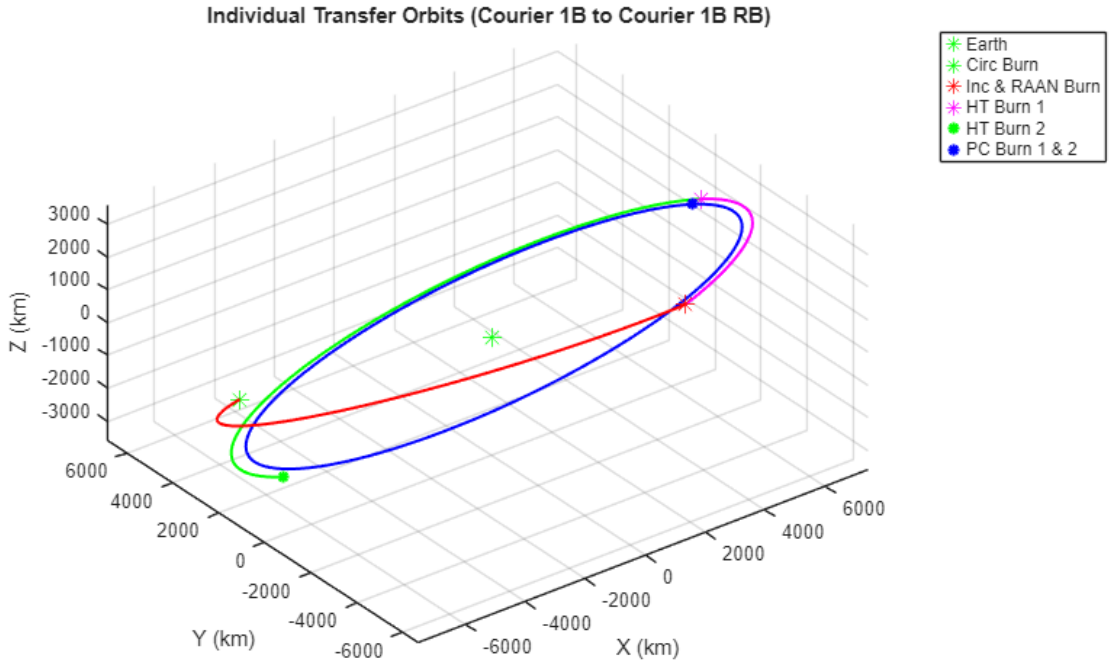
```
JDCurrent = JDCurrent + dt;

state2p = [RL2JD,VL2JD];
tspan = [0,dt];
[~,Orbit2C] = ode45(@twobodymotion,tspan,state2p,options,mu); %orbit 2 cont.
RL2JDC = [Orbit2C(end,1),Orbit2C(end,2),Orbit2C(end,3)];
VL2JDC = [Orbit2C(end,4),Orbit2C(end,5),Orbit2C(end,6)];
%these will be the same after 5 periods so can be reused
```

LEO to LEO Plots

```
figure('Name', 'Transfer Orbit Courier 1B to Courier 1B RB');
addEarth();
hold on;
plot3(Orbit1(:, 1), Orbit1(:, 2), Orbit1(:, 3), 'b', 'LineWidth', 1.5);
%Orbit 1
plot3(Orbit2(:, 1), Orbit2(:, 2), Orbit2(:, 3), 'g', 'LineWidth', 1.5);
%Orbit 2
plot3(Trans12_1(:, 1), Trans12_1(:, 2), Trans12_1(:, 3), 'r', 'LineWidth',
1.5); %Circ Orbit
plot3(Trans12_2(:,1),Trans12_2(:,2),Trans12_2(:,3), 'r', 'LineWidth', 1.5);
%Inc raan orbit
plot3(HT1(:, 1), HT1(:, 2), HT1(:, 3), 'r', 'LineWidth', 1.5); %HT Trans
orbit
plot3(Phase1(:, 1), Phase1(:, 2), Phase1(:, 3), 'r', 'LineWidth', 1.5);
%Phase change orbit
xlabel('X (km)');
ylabel('Y (km)');
zlabel('Z (km)');
grid on;
legend('Earth','LEO1 (Courier 1B)','LEO2 (Courier 1B RB)','Transfer Orbit')
title('Transfer Orbit (Courier 1B to Courier 1B RB)');
axis equal

figure('Name', 'Transfer Orbit Courier 1B to Courier 1B RB');
plot3(0, 0, 0, 'g*', 'MarkerSize', 10); % Earth at the origin
hold on;
plot3(Orbit1A(end,1),Orbit1A(end,2),Orbit1A(end,3), 'g*', 'MarkerSize', 10);
%Circ Burn
plot3(Trans12_1(end,1),Trans12_1(end,2),Trans12_1(end,3), 'r*',
'MarkerSize', 10); %Inc Raan burn
plot3(RL1CP(1),RL1CP(2),RL1CP(3),'m*', 'MarkerSize',10) %HT burn 1
plot3(HT1(end, 1), HT1(end, 2), HT1(end, 3), 'g*', 'LineWidth', 1.5); %HT
burn 2
plot3(Phase1(end, 1), Phase1(end, 2), Phase1(end, 3), 'b*', 'LineWidth',
1.5); %Both Phase change burns & Rendezvous
plot3(Trans12_1(:, 1), Trans12_1(:, 2), Trans12_1(:, 3), 'r', 'LineWidth',
1.5); %Circ Orbit
```

```
plot3(Trans12_2(:,1),Trans12_2(:,2),Trans12_2(:,3), 'm', 'LineWidth', 1.5);
%Inc raan orbit
plot3(HT1(:, 1), HT1(:, 2), HT1(:, 3), 'g', 'LineWidth', 1.5); %HT Trans
orbit
plot3(Phase1(:, 1), Phase1(:, 2), Phase1(:, 3), 'b', 'LineWidth', 1.5);
%Phase change orbit
xlabel('X (km)');
ylabel('Y (km)');
zlabel('Z (km)');
grid on;
legend('Earth','Circ Burn','Inc & RAAN Burn','HT Burn 1','HT Burn 2','PC
Burn 1 & 2')
title('Individual Transfer Orbits (Courier 1B to Courier 1B RB)');
axis equal
```





LEO to MEO

```
JDCurrent = JDCurrent + 5*pL2 + 0.5*pL2; %adding the 5.5 periods to time

%Use R2A and V2A state vectors (orbit 2 at apogee)

rT = norm(R2A); %R target

%Circ Burn
v = sqrt(mu/rT);
dVc2 = v-norm(V2A); %delta v from circ burn

uV = V2A/norm(V2A); %unit vector
dVd = dVc2*uV; %delta V with direction
VL2C = V2A+dVd; %new velocity vector after adding velocity
RL2C = R2A;
dVt = dVt + dVc2; %update total delta V

hL2C = cross(RL2C,VL2C); %h of Leo 1 circ orbit

[~,~,e] = rv2coes(RL2C,VL2C,mu,Rearth); %e confirms we are circ!


[RM,VM] = keplerian2ijk(aM*1000,eccM,incM,RAANM,ArgPM,nuM);
RM = RM./1000; %R&V at epoch of tle of meo object
VM = VM./1000;

[~,~,~,~,~,~,ArgPM] = rv2coes(RM,VM,mu,Rearth);
```

```
stateM = [RM,VM];
propTime2 = (JDStart-JDtimeM)+JDCurrent-(JDStart-JDtimeL1);
tspan = [0,propTime2];

[~,OrbitMS] = ode45(@twobodymotion,tspan,stateM,options,mu); %meo orbit
start after 5.5 periods with leo 2 orbit
RMS = [OrbitMS(end,1),OrbitMS(end,2),OrbitMS(end,3)];
VMS = [OrbitMS(end,4),OrbitMS(end,5),OrbitMS(end,6)];

%[~,~,~,~,~,~,~,pM,~,~,RaM,RpM] = rv2coes(RM,VM,mu,Rearth);

hM = cross(RMS,VMS);

nV = cross(hL2C,hM);
nVU = nV/norm(nV); %unit vector node line between leo2 circ orbit and meo orb

RM23B = norm(RL2C)*-nVU; %pos on leo2 circ orb to do inc+raan burn

%INC + RAAN Change
dR = abs(RAANL2-RAANM); %deg
alpha = acosd(cosd(incL2)*cosd(incM)+sind(incL2)*sind(incM)*cosd(dR));
dVincR = 2*norm(VL2C)*sind((alpha/2));

dVt = dVt + dVincR; %delta V total

V23u = cross(hM,RM23B)/norm(cross(hM,RM23B)); %unit vector vel
VM23B = norm(VL2C)*V23u; %velocity at position of inc raan burn (after)

%finding time of inc raan burn for correct propagation time
theta2 = acos((dot(RM23B,RL2C))/... %in radians!! Law of dot products?
    (norm(RL2C)*norm(RM23B))); %Angle between apoapse (circ burn) & inc raan
change
S2 = theta2*norm(RL2C); %arclength
tS2 = S2/norm(VL2C); %in seconds (Time from circ burn till node line/inc
burn)
JDCurrent = JDCurrent + tS2;

%Resetting JDCurrent to actually be time in seconds since midnight of the
%22nd:
JDCurrent = JDCurrent - (JDStart-JDtimeL1);
%Now JDCurrent is time since nov 22 after the inc raan burn

JDCurrent1 = JDCurrent - tS2; %Ignore, used as a test point (PASSED)
```

Propogating Circ burn 2 and Inc+RAAN burn 2

```
tspan = [0,tS2];
stateC = [RL2C,VL2C];
[~,OrbitC2] = ode45(@twobodymotion,tspan,stateC,options,mu); %circ burn of
leo2
% RMS = [OrbitC2(end,1),OrbitC2(end,2),OrbitC2(end,3)];
% VMS = [OrbitC2(end,4),OrbitC2(end,5),OrbitC2(end,6)];
```

```
[~,~,~,~,~,~,~,p5] = rv2coes(RM23B,VM23B,mu,Rearth); %arg of peri of the
circ orbit of leo 2

tspan = [0,0.25*p5];
stateIR2 = [RM23B,VM23B];%inc raan burn of leo2 not plotted just used for
calcs
[~,OrbitIR2] = ode45(@twobodymotion,tspan,stateIR2,options,mu);
RIR2 = [OrbitIR2(end,1),OrbitIR2(end,2),OrbitIR2(end,3)];
VIR2 = [OrbitIR2(end,4),OrbitIR2(end,5),OrbitIR2(end,6)];
```

Hohmann Transfer 2

```
[~,~,~,nuM23,~,~,ArgPT2,p5,t2] = rv2coes(RIR2,VIR2,mu,Rearth); %arg of peri
of the circ orbit of leo 2

Rp = norm(RM23B);
Ra = norm(RMS); %*-(RM23B/norm(RM23B)); %multiply by unit vector and neg to
flip dir
ecc = (Ra-Rp)/(Ra+Rp);
a = Rp/(1-ecc); %probably correct

[RH2_1,VH2_1] = coes2rvd(a,ecc,incM,RAANM,ArgPT2,nuM23,mu);


VM23B2 = VM23B + V23u*norm(VH2_1);

[~,~,~,nu,~,~,w,p4,t4] = rv2coes(RH2_1,VH2_1,mu,Rearth);
tP = (p5/2)-t4;

tspan = [0,0.5*p4];
stateH2 = [RH2_1,VH2_1];
[~,OrbitH2] = ode45(@twobodymotion,tspan,stateH2,options,mu); %hohmann t 2
RH2_2 = [OrbitH2(end,1),OrbitH2(end,2),OrbitH2(end,3)];
VH2_2 = [OrbitH2(end,4),OrbitH2(end,5),OrbitH2(end,6)];

tspan = [0,tP];
stateIR2 = [RM23B,VM23B]; %First half of the IR2 orbit
[~,OrbitIR2_2] = ode45(@twobodymotion,tspan,stateIR2,options,mu);
RIR22 = [OrbitIR2_2(end,1),OrbitIR2_2(end,2),OrbitIR2_2(end,3)];
VIR22 = [OrbitIR2_2(end,4),OrbitIR2_2(end,5),OrbitIR2_2(end,6)];

theta3 = acosd((dot(RIR22,RH2_1))/... %in radians!! Law of dot products?
    (norm(RIR22)*norm(RH2_1))); %Angle between apoapse (circ burn) & inc
raan change
dt = p5*(theta3/360);

tspan = [0,dt];
stateIR2 = [RIR22,VIR22]; %final part of the inc raan changed circ orb
[~,OrbitIR2_3] = ode45(@twobodymotion,tspan,stateIR2,options,mu);
RIR2_3 = [OrbitIR2_3(end,1),OrbitIR2_3(end,2),OrbitIR2_3(end,3)];
VIR2_3 = [OrbitIR2_3(end,4),OrbitIR2_3(end,5),OrbitIR2_3(end,6)];
```

```
JDCurrent = JDCurrent + tP + dt + 0.5*p4;

tspan = [0, JDCurrent+(JDStart-JDtimeM)];
[~,OrbitMS_2] = ode45(@twobodymotion,tspan,stateM,options,mu); %meo orbit
after transfers (now need to phase change)
RMS_2 = [OrbitMS_2(end,1),OrbitMS_2(end,2),OrbitMS_2(end,3)];
VMS_2 = [OrbitMS_2(end,4),OrbitMS_2(end,5),OrbitMS_2(end,6)];

[~,~,~,nut] = rv2coes(RMS_2,VMS_2,mu,Rearth);

VH2_2u = VH2_2/norm(VH2_2);
VMH2 = norm(VM)*VH2_2u;
```

Phase Change for Second Transfer

```
[~,~,~,nuM_2r,~,~,wt1,~,tp] = rv2coes(RMS_2,VMS_2,mu,Rearth); %true anom of
meo orbit
[~,~,~,nu2,~,~,wt2,pM,t4] = rv2coes(RH2_2,VMH2,mu,Rearth); %calculating our
position
[REH2,VEH2] = coes2rvd(aM,eccM,incM,RAANM,ArgPM,180,mu);
[RMP,VMP] = coes2rvd(aM,eccM,incM,RAANM,ArgPM,0,mu);
```

Calculating Delta V for Second Hohmann Transfer

```
dVB1 = VH2_1' - VIR2_3; %delta V from burn 1
dVB2 = VEH2' - VH2_2;
dVt = dVt + norm(dVB1) + norm(dVB2);
```

Propogation to Phase Change at perigee

```
stateEH = [REH2,VEH2];
tspan = [0, pM/2];
[~,OrbitMS_2P] = ode45(@twobodymotion,tspan,stateEH,options,mu); %our
position on meo orbit at perigee (after ht)
RMS_2P = [OrbitMS_2P(end,1),OrbitMS_2P(end,2),OrbitMS_2P(end,3)];
VMS_2P = [OrbitMS_2P(end,4),OrbitMS_2P(end,5),OrbitMS_2P(end,6)];

state32 = [RMS_2,VMS_2];
[~,OrbitMS_3] = ode45(@twobodymotion,tspan,state32,options,mu); %our
position on meo orbit at perigee (after ht)
RMS_3 = [OrbitMS_3(end,1),OrbitMS_3(end,2),OrbitMS_3(end,3)];
VMS_3 = [OrbitMS_3(end,4),OrbitMS_3(end,5),OrbitMS_3(end,6)];

JDCurrent = JDCurrent + pM/2;
```

Phase Change 2

```
[~,~,~,nuMP,~,~,wt1,~,tp1] = rv2coes(RMS_2P,VMS_2P,mu,Rearth); %true anom of
meo orbit
```

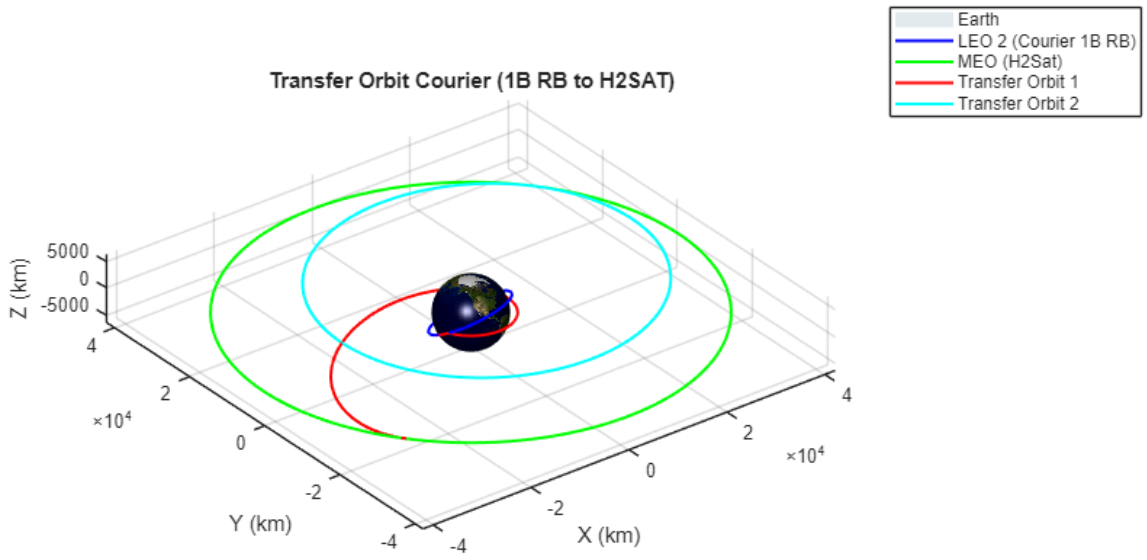
```
[~,~,~,nuJDC,~,~,wt1,~,tp] = rv2coes(RMS_3,VMS_3,mu,Rearth); %true anom of
meo orbit

% minD = inf;
% for i = 0:.001:360
%     [R,V] = coes2rvd(aM,eccM,incM,RAANM,ArgPM,i,mu);
%
%     dist = norm(RMS_3' - R);
%     if dist < minD
%         minD = dist;
%         deg = i;
%         R1 = R;
%     end
%
% end


%Orbit 1: (Ours)
Rp1 = norm(RM); %km
Ra1 = norm(RM); %km
theta1B = rad2deg(nuMP); %deg (of us)
theta1C = 127.368; %deg (of Meo) %Divide diff in angle by amount of periods
to catch up (will lower delta v)
ecc1 = eccM;
a1 = aM;
h2 = sqrt(mu*a1*(1-ecc1^2));
T1 = pM;

Rb1 = ((h2^2)/mu)/(1+ecc1*cosd(theta1B)); %current position of s/c B
Vb1t = (mu/h2)*(1+ecc1*cosd(theta1B)); %V tangential
Vb1r = (mu/h2)*ecc1*sind(theta1B); %V radial
Vb2t = (mu/h2)*(1+ecc1*cosd(theta1C)); %V tangential
Vb2r = (mu/h2)*ecc1*sind(theta1C); %V radial
Vt1 = sqrt((Vb1t^2)+(Vb1r^2));
FPAb1 = atand(Vb1r/Vb1t); %flight path angle in deg
FPAb2 = atand(Vb2r/Vb2t); %flight path angle in deg

%For us
E = 2*atan(sqrt((1-ecc1)/(1+ecc1))*tand(theta1B/2)); %eccentric anomaly
Me = E-ecc1*sin(E); %Mean anomaly
ts = (Me*T1)/(2*pi); %time since periapsis

%For Leo 2 (H2sat)
E2 = 2*atan(sqrt((1-ecc1)/(1+ecc1))*tand(theta1C/2)); %eccentric anomaly
Me2 = E2-ecc1*sin(E2); %Mean anomaly
ts2 = (Me2*T1)/(2*pi); %time since periapsis

%Orbit 2: (Phase Change Orb)
dt = (T1-(ts2-ts)); %total time to catch up by
%dt = dt1/10;
a2 = (dt^(2/3)*mu^(1/3))/(2^(2/3)*pi^(2/3));
Rp2 = norm(RM); %THIS NEEDED TO CHANGE
Ra2 = 2*a2-Rp2;
ecc2 = abs(Rp2-Ra2)/(Ra2+Rp2); %<3: is elliptical
```

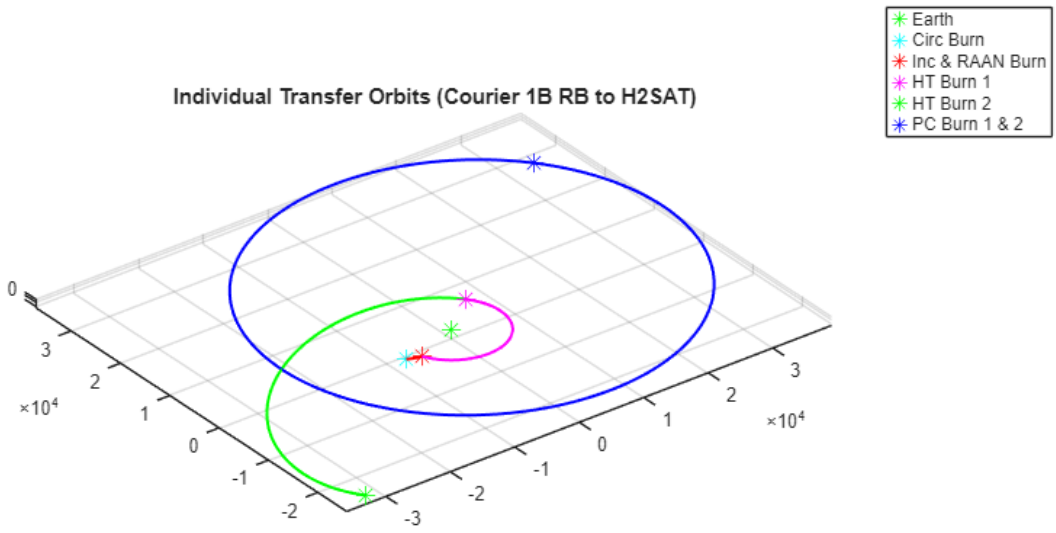
```
h2 = sqrt(mu*a2*(1-ecc2^2));
Vt2 = h2/Rp2;

%DeltaV EQ:
%2nd Flight path angle is 0:
dV = 2*sqrt((Vt1^2)+(Vt2^2)-2*Vt1*Vt2*cosd(FPAb2-FPAb1));
%This is both burns combined (each burn for PC is half of this)

dVt = dVt + dV;

[RPH2,VPH2] = coes2rvd(a2,ecc2,incM,RAANM,ArgPM,0,mu);

statePH2 = [-RPH2,-VPH2];
tspan = [0,dt];
[~,PH2] = ode45(@twobodymotion,tspan,statePH2,options,mu); %phase change
orbit
RPH2_2 = [PH2(end,1),PH2(end,2),PH2(end,3)];
VPH2_2 = [PH2(end,4),PH2(end,5),PH2(end,6)];

tspan = [0,dt];
state32 = [RMS_3,VMS_3];
[~,OrbitMS_4] = ode45(@twobodymotion,tspan,state32,options,mu); %our
position on meo orbit at perigee (after ht)
RMS_4 = [OrbitMS_4(end,1),OrbitMS_4(end,2),OrbitMS_4(end,3)]; %departure R&V
VMS_4 = [OrbitMS_4(end,4),OrbitMS_4(end,5),OrbitMS_4(end,6)];

JDCurrent = JDCurrent + dt;
JDCurrent = JDCurrent + 5*pM; %adding 5 periods orbiting with H2Sat
```

Plotting

```
figure('Name', 'Transfer Orbit Courier 1B RB to H2SAT');
addEarth();
hold on;
plot3(Orbit2(:, 1), Orbit2(:, 2), Orbit2(:, 3), 'b', 'LineWidth', 1.5);
%Orbit 2
plot3(OrbitMS(:, 1), OrbitMS(:, 2), OrbitMS(:, 3), 'g', 'LineWidth', 1.5);
%orbit 3
plot3(OrbitC2(:,1),OrbitC2(:,2),OrbitC2(:,3), 'r', 'LineWidth', 1.5); %circ
orbit 2
plot3(OrbitIR2_2(:,1),OrbitIR2_2(:,2),OrbitIR2_2(:,3), 'r', 'LineWidth',
1.5); %Inc raan orbit
plot3(OrbitIR2_3(:,1),OrbitIR2_3(:,2),OrbitIR2_3(:,3), 'r', 'LineWidth',
1.5); %Inc raan orbit
plot3(OrbitH2(:, 1), OrbitH2(:, 2), OrbitH2(:, 3), 'r', 'LineWidth', 1.5);
%HT Trans orbit
plot3(PH2(:, 1), PH2(:, 2), PH2(:, 3), 'c', 'LineWidth', 1.5);
xlabel('X (km)');
ylabel('Y (km)');
zlabel('Z (km)');
grid on;
legend('Earth','LEO 2 (Courier 1B RB)','MEO (H2Sat)','Transfer Orbit
1','','','','Transfer Orbit 2');
```

```
title('Transfer Orbit Courier (1B RB to H2SAT)');
axis equal

figure('Name', 'Transfer Orbit Courier 1B RB to H2SAT');
plot3(0, 0, 0, 'g*', 'MarkerSize', 10); % Earth at the origin\
hold on
grid on
plot3(OrbitC2(1,1),OrbitC2(1,2),OrbitC2(1,3),'c*', 'MarkerSize',10) %circ
burn
plot3(OrbitC2(end,1),OrbitC2(end,2),OrbitC2(end,3),'r*', 'MarkerSize',10)
%inc + raan burn
plot3(OrbitIR2_3(end,1),OrbitIR2_3(end,2),OrbitIR2_3(end,3),'m*',
'MarkerSize',10); %HT Burn 1
plot3(OrbitH2(end, 1), OrbitH2(end, 2), OrbitH2(end, 3),'g*',
'MarkerSize',10); %HT burn 2
plot3(RMP(1),RMP(2),RMP(3),'b*', 'MarkerSize',10);
plot3(OrbitC2(:,1),OrbitC2(:,2),OrbitC2(:,3), 'r', 'LineWidth', 1.5); %circ
orbit 2
plot3(OrbitIR2_2(:,1),OrbitIR2_2(:,2),OrbitIR2_2(:,3), 'm', 'LineWidth',
1.5); %Inc raan orbit
plot3(OrbitIR2_3(:,1),OrbitIR2_3(:,2),OrbitIR2_3(:,3), 'm', 'LineWidth',
1.5); %Inc raan orbit
plot3(OrbitH2(:, 1), OrbitH2(:, 2), OrbitH2(:, 3), 'g', 'LineWidth', 1.5);
%HT Trans orbit
plot3(PH2(:, 1), PH2(:, 2), PH2(:, 3), 'b', 'LineWidth', 1.5);
legend('Earth', 'Circ Burn','Inc & RAAN Burn','HT Burn 1','HT Burn 2','PC
Burn 1 & 2')
axis equal
title('Individual Transfer Orbits (Courier 1B RB to H2SAT)');
```

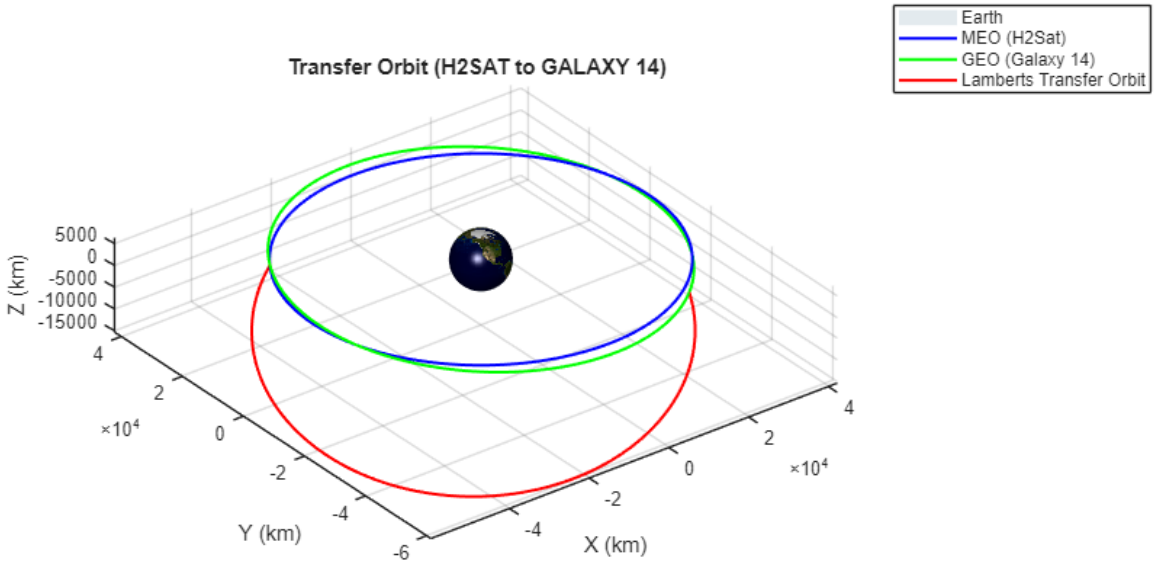


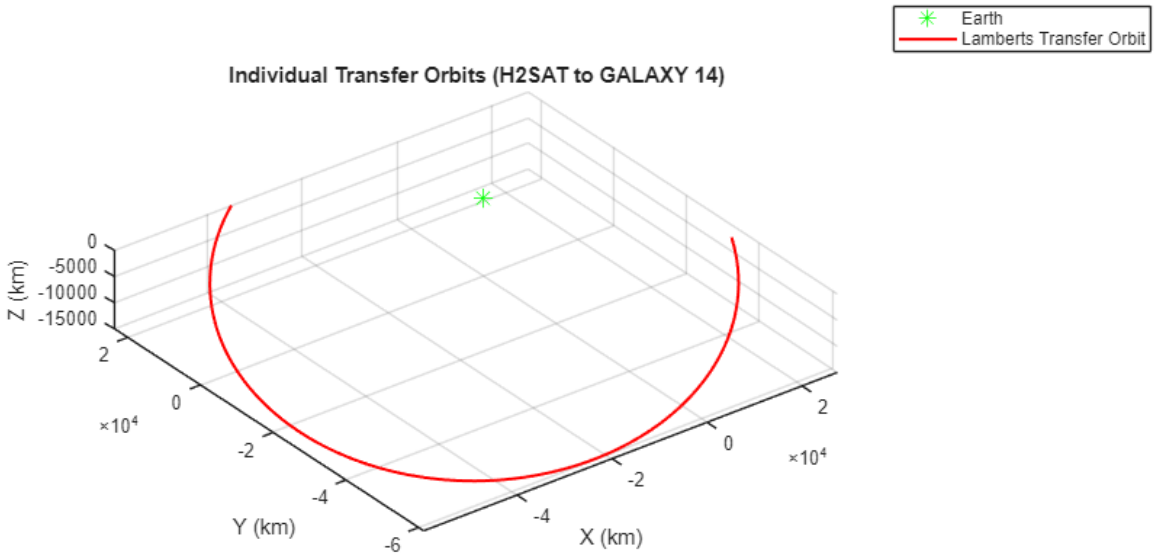


MEO to GEO

```
[RG,VG] = coes2rvd(aG,eccG,incG,RAANG,ArgPG,nuG,mu); %Starting geo R&V

tspan = [0,JDCurrent+(JDStart-JDtimeG)+78929];
stateGS = [RG,VG];
[~,OrbitGS] = ode45(@twobodymotion,tspan,stateGS,options,mu); %position of
geo orbit when done orbiting 5 times with meo
RGS = [OrbitGS(end,1),OrbitGS(end,2),OrbitGS(end,3)]; %departure R&V
VGS = [OrbitGS(end,4),OrbitGS(end,5),OrbitGS(end,6)];

[V1,V2] = lambUVBi(RMS_4,RGS,78929,1,mu,tol);

dV1 = V1 - VMS_4;
dV2 = VGS - V2;
dVL = norm(dV1) + norm(dV2);
dVt = dVt + dVL;

tspan = [0,78929];
stateLE = [RMS_4,V1];
[~,OrbitL] = ode45(@twobodymotion,tspan,stateLE,options,mu); %position of
geo orbit when done orbiting 5 times with meo
RLE = [OrbitL(end,1),OrbitL(end,2),OrbitL(end,3)]; %departure R&V
VLE = [OrbitL(end,4),OrbitL(end,5),OrbitL(end,6)];

% x = 1;
% for i = 3600:1:26*3600
%     dtL = i;
```

```
%     %Ttrnsfr = JDCurrent + dtL;
%     tspan = [0,dtL];
%     stateL2 = [RGS, VGS];
%
%     [~,stateNewG] = ode45(@twobodymotion,tspan,stateL2,options,mu);
%     RL2 = [stateNewG(end,1),stateNewG(end,2),stateNewG(end,3)];
%     VL2 = [stateNewG(end,4),stateNewG(end,5),stateNewG(end,6)];
%
%     %parameters for lambert w/ uv function:
%     tol = 1e-8;
%     tm = 1; %<3: Short way around
%
%     [V1,V2] = lambUVBi(RMS_4,RL2,dtL,tm,mu,tol);
%
%     Vf1(x,1:3) = V1 - VMS_4;
%     Vf1n(x) = norm(Vf1(x));
%     Vf2(x,1:3) = VL2 - V2;
%     Vf2n(x) = norm(Vf2(x));
%     Vf(x) = Vf1n(x) + Vf2n(x);
%     x = x +1;
% end
%
% disp([num2str(Vf)])
% disp([num2str(norm(Vf))])

[~,~,~,~,~,~,~,pG] = rv2coes(RGS,VGS,mu,Rearth);

figure('Name', 'Transfer Orbit H2SAT to GALAXY 14 (G-14)');
addEarth();
hold on;
plot3(OrbitMS(:, 1), OrbitMS(:, 2), OrbitMS(:, 3), 'b', 'LineWidth', 1.5);
plot3(OrbitGS(:, 1), OrbitGS(:, 2), OrbitGS(:, 3), 'g', 'LineWidth', 1.5);
plot3(OrbitL(:, 1), OrbitL(:, 2), OrbitL(:, 3), 'r', 'LineWidth', 1.5);
xlabel('X (km)');
ylabel('Y (km)');
zlabel('Z (km)');
legend('Earth','MEO (H2Sat)','GEO (Galaxy 14)','Lamberts Transfer Orbit')
title('Transfer Orbit (H2SAT to GALAXY 14)')
axis equal

figure('Name', 'Transfer Orbit H2SAT to GALAXY 14 (G-14)');
plot3(0, 0, 0, 'g*', 'MarkerSize', 10); % Earth at the origin
hold on;
grid on
plot3(OrbitL(:, 1), OrbitL(:, 2), OrbitL(:, 3), 'r', 'LineWidth', 1.5);
xlabel('X (km)');
ylabel('Y (km)');
zlabel('Z (km)');
legend('Earth','Lamberts Transfer Orbit')
title('Individual Transfer Orbits (H2SAT to GALAXY 14)')
axis equal
```





Final Display

```
disp(['Total Delta V (km/s): ', num2str(dVt)])
```

Total Delta V (km/s): 13.1136

Functions

```
function [a] = Me2a(Me,mu)
    %ME2A Finds Semi-Major Axis from Mean Motion
    %   [a] = Me2a(Me,mu)

    Me1 = Me*((2*pi)/86400); %Converts from rev/day to rad/s
    a = (mu/(Me1^2))^(1/3); %finds Semi-Major axis (a)
end

function earthHandle = addEarth(radius, axHandle)
    % addEarth - Generates a 3D Earth with corrected orientation and adds it
to a specified axes.
    %
    % Syntax:
    %   earthHandle = addEarth(radius, axHandle)
    %
    % Inputs:
    %   radius - Radius of the Earth (default: 6371 km).
    %   axHandle - Axes handle where Earth will be plotted (default: gca).
    %
    % Outputs:
    %   earthHandle - Handle to the Earth surface object.

    if nargin < 1
        radius = 6378; % Default Earth's radius in kilometers
    end
    if nargin < 2
        axHandle = gca; % Default to the current axes
    end

    % Load Earth texture
    earthTexture = imread('https://eoimages.gsfc.nasa.gov/images/
imagerecords/57000/57730/land_ocean_ice_2048.png');
    earthTexture = flipud(earthTexture); % Flip texture vertically to
correct poles

    % Create a sphere
    [lon, lat, Z] = sphere(50); % Higher resolution for smoother appearance

    % Scale the sphere to the specified radius
    X = radius * lon;
    Y = radius * lat;

    % Create surface object
    earthHandle = surf(axHandle, X, Y, radius * Z, 'EdgeColor', 'none', ...
        'FaceColor', 'texturemap', 'CData', earthTexture);

    % Adjust lighting and material properties
    lightangle(axHandle, -45, 30);
    lighting(axHandle, 'phong');
```

```
    material(axHandle, 'shiny');
    axis(axHandle, 'equal');
    %axis(axHandle, 'off'); % Optional: Hide axes for a cleaner look
end
```

Published with MATLAB® R2024b