Trajectory Analysis and Orbital Element Comparison for Bodies in Mars' Orbit

Conry Cabantac

AE313, Section 03DB, Roberto Cuellar

Embry-Riddle Aeronautical University, Daytona Beach Fl.

April 2024

Abstract

This study investigates the trajectory of a space station orbiting Mars over a one-year period, utilizing the two-body equations of motion. Additionally, the trajectories of Mars' moons, Phobos and Deimos, are plotted using SPICE software, incorporating the relevant parameters and kernels. Orbital elements for the space station, Phobos, and Deimos at their initial points are calculated using both analytical methods and SPICE software for comparison. The analysis shows that the equations used and the SPICE software yield very similar results with the highest percent difference being 1.72% between any two values.

Purpose

The purpose of this study is to analyze the trajectory of a space station orbiting Mars and compare it with the trajectories of Mars' moons over a one-year period. Using both analytical methods and SPICE software, the accuracy and reliability of each method in predicting the motion of each body is evaluated. The determination of the orbital elements for the space station and the moons serves to uncover any discrepancies between the analytical and SPICE-derived values.

Procedure

To perform this study, MATLAB is utilized to solve the two-body equation for the space station. Initially the given values for the space station are input in MATLAB along with the standard gravitational parameter of Mars. The initial conditions are given as

$$r = [2.042768482340556e + 4; 0; 0]km$$

and $v = [0; 1.447958841235277; 0]\frac{km}{s}$.

MATLAB's ODE45 is then used to solve and plot the trajectory of the space station's orbit over a one-year period beginning Jan

1st, 2024, and ending Dec 31st, 2024. Where the two-body equation of motion is defined by Equation 1.

$$\ddot{\underline{r}} = \frac{-\mu}{r^3}\underline{r} \qquad (1)$$

Following this the orbital elements are calculated using the following analytical methods along with SPICE's "cspice_oscelt" function.

First it is necessary to calculate the angular momentum of the orbit which is calculated using Equation 2.

$$\underline{h} = \underline{r} \times \underline{v} \qquad (2)$$

Where $\underline{r}$ is the position vector and $\underline{v}$ is the velocity vector.

Next the semimajor axis is calculated using Equation 3.

$$a = \frac{-\mu}{2(\frac{v^2}{2} - \frac{\mu}{r})} \qquad (3)$$

Where μ is the standard gravitational parameter of Mars, v is the magnitude of the velocity, and r is the magnitude of the position.

The inclination is calculated using Equation 4.

$$i = cos^{-1}(\frac{h_z}{h}) \qquad (4)$$

Where h_z is the z-component of the angular momentum and h is the magnitude of the angular momentum.

The next orbital element, RAAN is calculated using Equation 5.

$$\Omega = cos^{-1}\left(\frac{N_x}{N}\right) \text{ if } N_y \geq 0 \qquad (5)$$

$$\Omega = 360 - cos^{-1}\left(\frac{N_x}{N}\right) \text{ if } N_y < 0$$

Where $N = \sqrt{\widehat{N} \cdot \widehat{N}}$, $\widehat{N} = \widehat{I_z} \times \underline{h}$, $\widehat{I_z} = [0; 0; 1]$, N_x is the x-component of $\widehat{N}$, and N_y is the y-component of $\widehat{N}$.

Next the eccentricity is calculated using Equation 6.

$$\underline{e} = \frac{\underline{v} \times \underline{h}}{\mu} - \frac{\underline{r}}{r} \qquad (6)$$

Where e is the magnitude of the eccentricity vector $\underline{e}$.

The argument of perigee is calculated using Equation 7.

$$\omega = cos^{-1}(\frac{\widehat{N} \cdot \underline{e}}{N \times e}) \text{ if } e_z \geq 0 \qquad (7)$$

$$\omega = 360 - cos^{-1}(\frac{\widehat{N} \cdot \underline{e}}{N \times e}) \text{ if } e_z < 0 \qquad (7)$$

Where e_z is the z-component of the eccentricity vector.

Finally, the true anomaly is calculated using Equation 8.

$$\theta = cos^{-1}(\frac{1}{e} \cdot \frac{h^2}{\mu \cdot r} - 1) \text{ if } v_r \geq 0 \qquad (8)$$

$$\theta = cos^{-1}(\frac{1}{e} \cdot \frac{h^2}{\mu \cdot r} - 1) \text{ if } v_r < 0 \qquad (8)$$

Where $v_r = \frac{r \cdot v}{r}$.

For the moons, the respective kernels are used, including "de430.bsp", "mar097.bsp" and "naif0012.tls.pc", for their respective initial conditions. SPICE is then configured to plot the trajectories of Mars' moons. The frame used is MARSIAU, there are no aberration corrections, and the observer is Mars Barycenter. Lastly the orbital elements are calculated using analytical methods along with SPICE's "cspice_oscelt" function and the two are compared using Equation 9 to calculate the percent difference.

$$\% \, diff = \frac{|C_{SPICE} - C_{eq}|}{C_{eq}} * 100\% \qquad (9)$$

Where C_{SPICE} is the value obtained from the SPICE software and C_{eq} is the value obtained from the equations.

Results

After computing the trajectories and the orbital elements for all the relevant bodies, a plot is obtained, and an analysis can be conducted on the accuracy of both methods and the discrepancies can be identified.

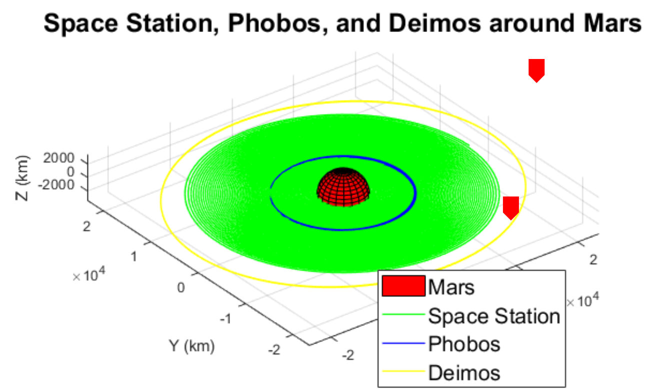

Figure 1: Plot of the trajectories of the space station, Phobos, and Deimos around Mars.

Space Station	Equations	SPICE	% Diff
Eccentricity	6.1593e-07	6.1593e-07	0
RAAN (deg)	0	0	0
Inclination (deg)	0	0	0
Argument of Perigee (deg)	180	180	0
Mean Anomaly (deg)	179.998	180	0.0011
Semimajor Axis (km)	20427.7	20427.7	0

Figure 2: Table displaying the values obtained for the Space station including all classic orbital elements compared to the SPICE values.

Phobos	Equations	SPICE	% Diff
Eccentricity	0.0151741	0.0151741	0
RAAN (deg)	309.117	309.117	0
Inclination (deg)	1.08939	1.08939	0
Argument of Perigee (deg)	289.562	289.562	0
Mean Anomaly (deg)	265.126	266.86	0.654
Semimajor Axis (km)	9398.92	9236.69	1.72

Figure 3: Table displaying the values obtained for Phobos including all classic orbital elements compared to the SPICE values.

Deimos	Equations	SPICE	% Diff
Eccentricity	0.000336311	0.000336311	0
RAAN (deg)	245.024	245.024	0
Inclination (deg)	2.58693	2.58693	0
Argument of Perigee (deg)	168.635	168.635	0
Mean Anomaly (deg)	311.802	311.83	0.00898
Semimajor Axis (km)	23448.1	23450.7	0.0111

Figure 4: Table displaying the values obtained for Deimos including all classic orbital elements compared to the SPICE values.

Conclusion

After completing the study and analysis of the data, the largest percentage difference between any of the values was 1.72%. This shows that the SPICE functions and the equations used to calculate the orbital elements were successful in terms of accuracy as the results show they were very close to one another. It should be noted that the mean anomaly and semimajor axis each had some percent difference of less than one percent but are not entirely similar. It should also be noted that the plot of the space station is incorrect as it shows a disk or orbit rather than one singular path. Although the plot is incorrect, the orbital elements showed little to no discrepancies compared to the SPICE analysis meaning that the issue of the plot has to do with the MATLAB code rather than the values shown in Appendix 1.

<u>Appendix</u>

Appendix 1: MATLAB Script and Outputs

```
%{
Name: Conry Cabantac
Email:cabantac@my.erau.edu
Date:4/17/24
AE 313 Section 03DB
Assignment: AE 313 SPICE Project
Program Description: Use SPICE to plot a stations trajectory and the moons around
Mars.
Worked With: Alone
%}

clc;
clear;
close all;

global mu

% Space Station Initial Conditions %
r = [2.042768482340556e+04;0;0]; % Km
r_mag = norm(r);
v = [0;1.447958841235277;0]; % Km/s
v_mag = norm(v);
deg = pi/180;

r0 = [2.042768482340556e+04;0;0;0;1.447958841235277;0]; % State Vector at t0

% Mars' Standard Gravitational Parameter
mu = 4.28284e+04; % km^3/s^2

% Spice %

% Add Paths
addpath('C:\Users\Conry\Documents\Spice project AE313\mice\src\mice')
addpath('C:\Users\Conry\Documents\Spice project AE313\mice\lib')
addpath('C:\Users\Conry\Documents\Spice project AE313\mice\kernels')

% Clear Previous Kernals
cspice_kclear

% Load Kernels (via metakernel)
cspice_furnsh('C:\Users\Conry\Documents\Spice project AE313\mice\metakernel.txt')

frame = 'MARSIAU';
abcorr = 'None'; % Aberration corrections
observer = '4'; % Mars Barycenter

% Simulation Times
et_initial = cspice_str2et('Jan 1 2024'); % Initial Epoch
et_final = cspice_str2et('Dec 31 2024');  % Final Epoch
tf = 1*365*86400;
h = 1000;
% times = (et_initial(1):h:et_initial(1)+tf);
times = (et_initial(1):h:et_final(1));
tspan = linspace(times(1),times(end),31537);
```

```
% Numerical Integration
[t,y] = ode45(@rhs, tspan, r0);

xout=y(:,1);
yout=y(:,2);
zout=y(:,3);
vxout=y(:,4);
vyout=y(:,5);
vzout=y(:,6);

% Orbital Elements Space Station
h = cross(r,v); % Angular Momentum vector
h_mag = norm(h); % Angular Momentum
Iz_hat = [0 0 1]; % Z Direction Unit Vector

% Eccentricity
e = ((cross(v,h)/mu))-(r/r_mag); % Eccentricity Vector
e_mag = norm(e); % Eccentricity

% Semi major axis
a = mu/(2*((v_mag^2)/2)); % Km

% Inclination
inclination = acos(dot(h,Iz_hat)/h_mag);

% RAAN
N_hat = cross(Iz_hat,h);
N_mag = norm(N_hat);
Nx = N_hat(1,1);
Ny = N_hat(1,2);
if isnan(Nx/N_mag)
    if Ny >= 0
        RAAN = (acos(1)/deg);
        elseif Ny < 0
        RAAN = 360 - (acos(1)/deg);
    end
else
    if Ny >= 0
        RAAN = (acos(Nx/N_mag)/deg);
        elseif Ny < 0
        RAAN = 360 - (acos(Nx/N_mag)/deg);
    end
end

% Argument of Perigee
ez = e(3,1);
if isnan(dot(N_hat,e)/(N_mag*e_mag))
    if ez >= 0
        argument_of_perigee = (acos(-1))/deg;
        elseif ez < 0
        argument_of_perigee = 360 - (acos(-1)/deg);
    end
else
    if ez >= 0
        argument_of_perigee = (acos(dot(N_hat,e)/(N_mag*e_mag)))/deg;
        elseif ez < 0
        argument_of_perigee = 360 - (acos(dot(N_hat,e)/(N_mag*e_mag))/deg);
    end
end

% True Anomoly
```

```
vr = (dot(r,v)/r_mag);
if vr >= 0
    true_anomoly = (acos((1/e_mag)*((h_mag^2)/(mu*r_mag)-1)))/deg;
    elseif vr < 0
    true_anomoly = 360 - (acos((1/e_mag)*((h_mag^2)/(mu*r_mag)-1))/deg);
end

% Plot orbit and spherical Mars in 3D
[x,y,z]=sphere; % Generate Mars as Sphere
x=x*3390;y=y*3390;z=z*3390; % Scale to Mars Radius
figure(1)
mars = surf(x,y,z); % Plot Mars
set(mars,'FaceColor',[1 0 0]) % Red
hold on
plot3(xout,yout,zout,'LineWidth',1,'Color',[0 1 0]) % Green Orbit
axis equal
xlabel('X (km)')
ylabel('Y (km)')
zlabel('Z (km)')
hold on


% Get Ephemeris
r_p = cspice_spkpos('Phobos',times,frame,'NONE', observer);   % Phobos Position
r_d = cspice_spkpos('Deimos',times,frame,'NONE', observer);   % Deimos Position

% Plot Phobos and Deimos orbits
plot3(r_p(1,:), r_p(2,:),r_p(3,:), 'LineWidth',1,'Color',[0 0 1]) % Blue Orbit
hold on
plot3(r_d(1,:), r_d(2,:),r_d(3,:), 'LineWidth',1,'Color',[1 1 0]) % Yellow Orbit
hold on
legend('Mars','Space Station','Phobos','Deimos','FontSize',15,Location='best')
title('Space Station, Phobos, and Deimos around Mars', 'FontSize', 18)

% COE of Space Craft Using SPICE Function
coe_sc = cspice_oscelt(r0, et_initial(1), mu);

% COE of Phobos Using SPICE Function
et_desired = cspice_str2et('Jan 1 2024');    % Initial Time
Phobos_state = cspice_spkezr('Phobos',et_desired,frame,abcorr,observer);

r_phobos = [Phobos_state(1,1) Phobos_state(2,1) Phobos_state(3,1)];
r_mag_phobos = norm(r_phobos);
v_phobos = [Phobos_state(4,1) Phobos_state(5,1) Phobos_state(6,1)];
v_mag_phobos = norm(v_phobos);

Pho_state = [r_phobos v_phobos]';
coe_pho = cspice_oscelt(Pho_state, et_desired, mu);


% COE of Phobos Using Given Equations
h_phobos = cross(r_phobos,v_phobos); % Angular Momentum vector
h_mag_phobos = norm(h_phobos); % Angular Momentum
Iz_hat = [0 0 1]; % Z Direction Unit Vector

% Eccentricity
e_phobos = ((cross(v_phobos,h_phobos)/mu))-(r_phobos/r_mag_phobos); % Eccentricity
Vector
e_mag_phobos = norm(e_phobos); % Eccentricity

% Semi major axis
a_phobos = mu/(2*((v_mag_phobos^2)/2)); % Km
```

```
% Inclination
inclination_phobos = acos((h_phobos(1,3))/h_mag_phobos);

% RAAN
N_hat_phobos = cross(Iz_hat,h_phobos);
N_mag_phobos = norm(N_hat_phobos);
Nx_phobos = N_hat_phobos(1,1);
Ny_phobos = N_hat_phobos(1,2);
if Ny_phobos >= 0
    RAAN_phobos = (acos(Nx_phobos/N_mag_phobos)/deg);
    elseif Ny_phobos < 0
    RAAN_phobos = 360 - (acos(Nx_phobos/N_mag_phobos)/deg);
end

% Argument of Perigee
ez_phobos = e_phobos(1,1);
if ez_phobos >= 0
    argument_of_perigee_phobos =
(acos(dot(N_hat_phobos,e_phobos)/(N_mag_phobos*e_mag_phobos)))/deg;
    elseif ez_phobos < 0
    argument_of_perigee_phobos = 360 -
(acos(dot(N_hat_phobos,e_phobos)/(N_mag_phobos*e_mag_phobos))/deg);
end

% Mean Anomoly
vr_phobos = (dot(r_phobos,v_phobos)/r_mag_phobos);
if vr_phobos >= 0
    true_anomoly_phobos = (acos((1/e_mag_phobos)*((h_mag_phobos^2)/(mu*r_mag_phobos)-
1)))/deg;
    elseif vr_phobos < 0
    true_anomoly_phobos =  360 -
(acos((1/e_mag_phobos)*((h_mag_phobos^2)/(mu*r_mag_phobos)-1))/deg);
end


% COE of Deimos Using SPICE Function
et_desired = cspice_str2et('Jan 1 2024');     % Initial Time
Deimos_state = cspice_spkezr('Deimos',et_desired,frame,abcorr,observer);

r_deimos = [Deimos_state(1,1) Deimos_state(2,1) Deimos_state(3,1)];
r_mag_deimos = norm(r_deimos);
v_deimos = [Deimos_state(4,1) Deimos_state(5,1) Deimos_state(6,1)];
v_mag_deimos = norm(v_deimos);

Dei_state = [r_deimos v_deimos]';
coe_dei = cspice_oscelt(Dei_state, et_desired, mu);
deg = pi/180;


% COE of Deimos Using Given Equations
h_deimos = cross(r_deimos,v_deimos); % Angular Momentum vector
h_mag_deimos = norm(h_deimos); % Angular Momentum
Iz_hat = [0 0 1]; % Z Direction Unit Vector

% Eccentricity
e_deimos = ((cross(v_deimos,h_deimos)/mu))-(r_deimos/r_mag_deimos); % Eccentricity
Vector
e_mag_deimos = norm(e_deimos); % Eccentricity

% Semi major axis
a_deimos = mu/(2*((v_mag_deimos^2)/2)); % Km

% Inclination
```

```
inclination_deimos = acos(dot(h_deimos,Iz_hat)/h_mag_deimos);

% RAAN
N_hat_deimos = cross(Iz_hat,h_deimos);
N_mag_deimos = norm(N_hat_deimos);
Nx_deimos = N_hat_deimos(1,1);
Ny_deimos = N_hat_deimos(1,2);
if Ny_deimos >= 0
    RAAN_deimos = (acos(Nx_deimos/N_mag_deimos)/deg);
    elseif Ny_deimos < 0
    RAAN_deimos = 360 - (acos(Nx_deimos/N_mag_deimos)/deg);
end

% Argument of Perigee
ez_deimos = e_deimos(1,1);
if ez_deimos >= 0
    argument_of_perigee_deimos =
(acos(dot(N_hat_deimos,e_deimos)/(N_mag_deimos*e_mag_deimos))/deg);
    elseif ez_deimos < 0
    argument_of_perigee_deimos = 360 -
(acos(dot(N_hat_deimos,e_deimos)/(N_mag_deimos*e_mag_deimos))/deg);
end

% Mean Anomoly
vr_deimos = (dot(r_deimos,v_deimos)/r_mag_deimos);
if vr_deimos >= 0
    true_anomoly_deimos = (acos((1/e_mag_deimos)*((h_mag_deimos^2)/(mu*r_mag_deimos)-
1)))/deg;
    elseif vr_deimos < 0
    true_anomoly_deimos = 360 -
(acos((1/e_mag_deimos)*((h_mag_deimos^2)/(mu*r_mag_deimos)-1))/deg);
end



% Display Results of COE %
fprintf('\n Space Station Orbital Elements at a Given Epoch Using SPICE function')
disp(' ')
fprintf('\n Eccentricity                            = %g', coe_sc(2))
fprintf('\n Right ascension (deg)                   = %g', coe_sc(4)/deg)
fprintf('\n Inclination (deg)                       = %g', coe_sc(3)/deg)
fprintf('\n Argument of perigee (deg)               = %g', coe_sc(5)/deg)
fprintf('\n Mean anomaly at epoch (deg)             = %g', coe_sc(6)/deg)
fprintf('\n Semimajor Axis (km)                     = %g\n\n', coe_sc(1))

% Display Results of COE %
fprintf('\n Space Station Orbital Elements at a Given Epoch Using Given Equations')
disp(' ')
fprintf('\n Eccentricity                            = %g', e_mag)
fprintf('\n Right ascension (deg)                   = %g', RAAN)
fprintf('\n Inclination (deg)                       = %g', inclination)
fprintf('\n Argument of perigee (deg)               = %g', argument_of_perigee)
fprintf('\n Mean anomaly at epoch (deg)             = %g', true_anomoly)
fprintf('\n Semimajor Axis (km)                     = %g\n\n', a)

fprintf('\n Phobos Orbital Elements at a Given Epoch Using SPICE function')
disp(' ')
fprintf('\n Eccentricity                            = %g', coe_pho(2))
fprintf('\n Right ascension (deg)                   = %g', coe_pho(4)/deg)
fprintf('\n Inclination (deg)                       = %g', coe_pho(3)/deg)
fprintf('\n Argument of perigee (deg)               = %g', coe_pho(5)/deg)
fprintf('\n Mean anomaly at epoch (deg)             = %g', coe_pho(6)/deg)
fprintf('\n Semimajor Axis (km)                     = %g\n\n', coe_pho(1))
```

```
fprintf('\n Phobos Orbital Elements at a Given Epoch Using Given Equations')
disp(' ')
fprintf('\n Eccentricity                              = %g', e_mag_phobos)
fprintf('\n Right ascension (deg)                     = %g', RAAN_phobos)
fprintf('\n Inclination (deg)                         = %g', inclination_phobos/deg)
fprintf('\n Argument of perigee (deg)                 = %g', argument_of_perigee_phobos)
fprintf('\n Mean anomaly at epoch (deg)               = %g', true_anomoly_phobos)
fprintf('\n Semimajor Axis (km)                       = %g\n\n', a_phobos)

fprintf('\n Deimos Orbital Elements at a Given Epoch Using SPICE function')
disp(' ')
fprintf('\n Eccentricity                              = %g', coe_dei(2))
fprintf('\n Right ascension (deg)                     = %g', coe_dei(4)/deg)
fprintf('\n Inclination (deg)                         = %g', coe_dei(3)/deg)
fprintf('\n Argument of perigee (deg)                 = %g', coe_dei(5)/deg)
fprintf('\n Mean anomaly at epoch (deg)               = %g', coe_dei(6)/deg)
fprintf('\n Semimajor Axis (km)                       = %g\n\n', coe_dei(1))

fprintf('\n Deimos Orbital Elements at a Given Epoch Using Given Equations')
disp(' ')
fprintf('\n Eccentricity                              = %g', e_mag_deimos)
fprintf('\n Right ascension (deg)                     = %g', RAAN_deimos)
fprintf('\n Inclination (deg)                         = %g', inclination_deimos/deg)
fprintf('\n Argument of perigee (deg)                 = %g', argument_of_perigee_deimos)
fprintf('\n Mean anomaly at epoch (deg)               = %g', true_anomoly_deimos)
fprintf('\n Semimajor Axis (km)                       = %g\n\n', a_deimos)




% ODE45
function dstatedt = rhs(t,f)
global mu

x = f(1);
y = f(2);
z = f(3);
vx = f(4);
vy = f(5);
vz = f(6);

r = norm([x y z]);

ax = -(mu*x)/r^3;
ay = -(mu*y)/r^3;
az = -(mu*z)/r^3;

dstatedt = [vx;vy;vz;ax;ay;az];

end
```

```
Warning: Failure at t=7.608936e+08.  Unable to meet integration tolerances

without reducing the step size below the smallest value allowed (1.907349e-06)

at time t.



 Space Station Orbital Elements at a Given Epoch Using SPICE function
```

```
Eccentricity                          = 6.1593e-07

Right ascension (deg)                 = 0

Inclination (deg)                     = 0

Argument of perigee (deg)             = 180

Mean anomaly at epoch (deg)           = 180

Semimajor Axis (km)                   = 20427.7



Space Station Orbital Elements at a Given Epoch Using Given Equations


Eccentricity                          = 6.1593e-07

Right ascension (deg)                 = 0

Inclination (deg)                     = 0

Argument of perigee (deg)             = 180

Mean anomaly at epoch (deg)           = 179.998

Semimajor Axis (km)                   = 20427.7



Phobos Orbital Elements at a Given Epoch Using SPICE function


Eccentricity                          = 0.0151741

Right ascension (deg)                 = 309.117

Inclination (deg)                     = 1.08939

Argument of perigee (deg)             = 289.562

Mean anomaly at epoch (deg)           = 266.86
```

Semimajor Axis (km) = 9236.69

Phobos Orbital Elements at a Given Epoch Using Given Equations

Eccentricity = 0.0151741

Right ascension (deg) = 309.117

Inclination (deg) = 1.08939

Argument of perigee (deg) = 289.562

Mean anomaly at epoch (deg) = 265.126

Semimajor Axis (km) = 9398.92

Deimos Orbital Elements at a Given Epoch Using SPICE function

Eccentricity = 0.000336311

Right ascension (deg) = 245.024

Inclination (deg) = 2.58693

Argument of perigee (deg) = 168.635

Mean anomaly at epoch (deg) = 311.83

Semimajor Axis (km) = 23450.7

Deimos Orbital Elements at a Given Epoch Using Given Equations

Eccentricity = 0.000336311

```
Right ascension (deg)                = 245.024

Inclination (deg)                    = 2.58693

Argument of perigee (deg)            = 168.635

Mean anomaly at epoch (deg)          = 311.802

Semimajor Axis (km)                  = 23448.1
```