Orbital Mechanics Project - Mars Sample Return

By: Harrison Perone

Date: 4/28/25

Overview

This project aims to calculate all of the orbital maneuvers required to complete the "Orbiter" half of the NASA / ESA Mars Sample Return mission. This means reaching Earth orbit, transferring to Mars, being captured into orbit around Mars, rendezvousing with the sample canister, and escaping Mars orbit to get back to Earth. In this context, a maneuver is a delta-V vector applied at a specific time. All the maneuvers were calculated with MATLAB scripts and functions. Simulink simulations corresponding to the different phases of the mission were used to verify the correctness of these maneuvers.

Definitions

Coordinate Systems

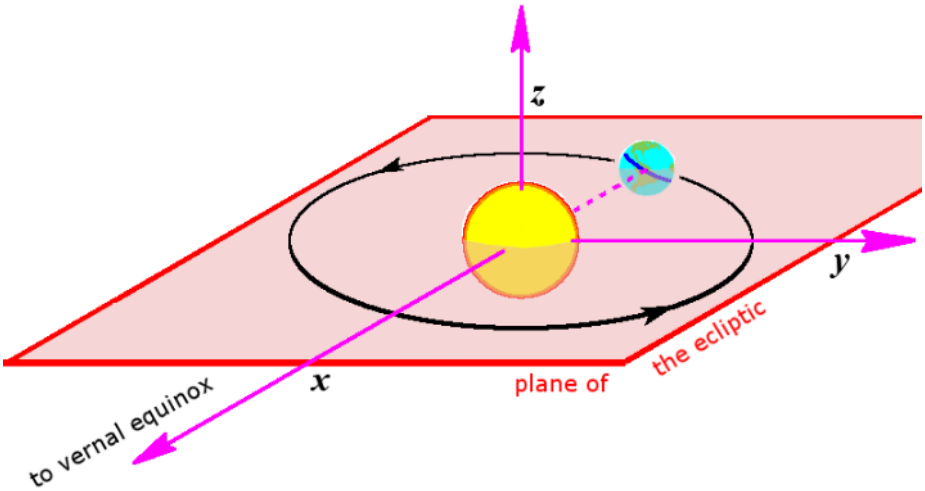
If the name of a coordinate system starts with Object-centered, that means that its origin is located at the center of that object and that its axes don't rotate relative to the background stars. However, if its name starts with Object-fixed, its origin is the same but its axes are fixed relative to the surface of the object.

Ecliptic Coordinates

Positive x: Points in the direction of the Sun (as seen from Earth) on the Vernal Equinox. In other words, it's the point on the night sky that defines 0 right assencion.

Positive z: In the direction of Earth's orbital angular momentum vector.

Positive y: Completes right handed triad.



Earth-fixed Equatorial

Positive x: Points in the direction of the Prime Meridian (0 degrees longitude)

Positive z: Points towards the north pole

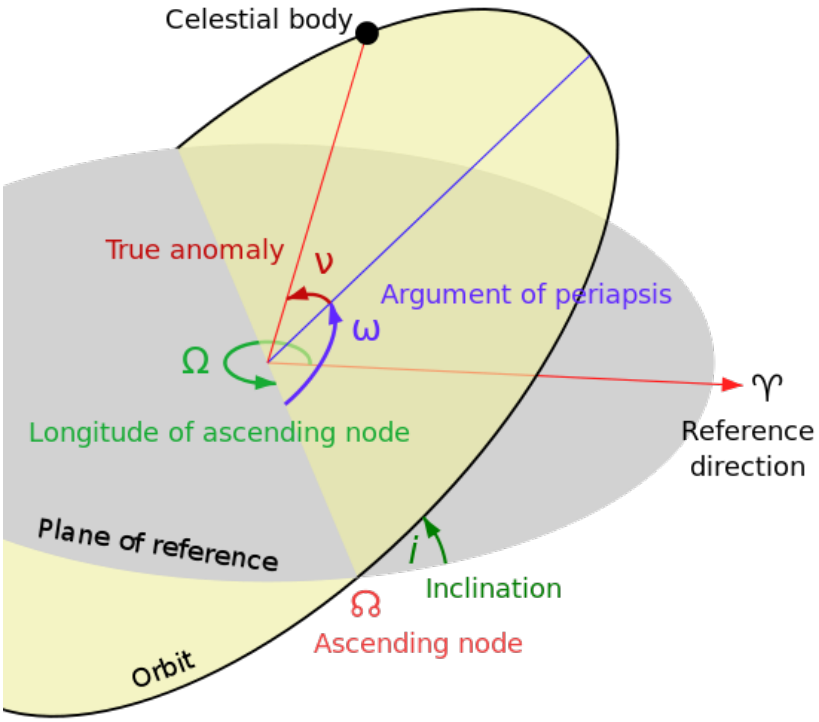
Positive y: Completes right handed triad.

Keplerian Orbital Elements

Used to define the size, shape, and orinetation of an orbit

- h - specific angular momentum
- i - inclination
- Ω - right ascension of the ascending node (RA)
- e - eccentricity
- ω - argument of perigee
- υ - true anomaly

Note that specific angular momentum is found by dividing the angular mometum of an obiting object by its mass. The elements for the orientation of the orbit are visualized in the diagram below.



Note that the reference direction is often defined to be the positive x axis of the coordinate system in use.

Mission Phases

Transfers Between Earth and Mars

Introduction

An interplanetary trajectory is a heliocentric orbit that takes a spacecraft from one planet to another. For this project, the transfers from Earth to Mars and vice versa need to be calculated. Although neither of these steps is first chronologically, it makes sense to start with them since the launch to Earth orbit requires the information about the Earth to Mars transfer.

Simplifying Assumptions

Assume Earth and Mars follow circular orbits around the Sun at their mean orbital radii and that their orbits lie in the same plane. Given the sizes of Earth and Mars relative to their orbits, this assumption would lead to large errors in real life. However, its results can still be used for mission planning purposes. Additionally, this section doesn't model the gravitational effect of any bodies other than the Sun since they are negligible for almost all of the transfer.

Problem Statement

Given the initial angles of Earth and Mars relative to the positive x axis (in Solar-centered Ecliptic coordinates), find the impulse maneuvers required to transfer from one planet to the other. Fuel usage should be minimized, and as such, the spacecraft can wait at the planet of origin until its transfer window arrives.

Solution

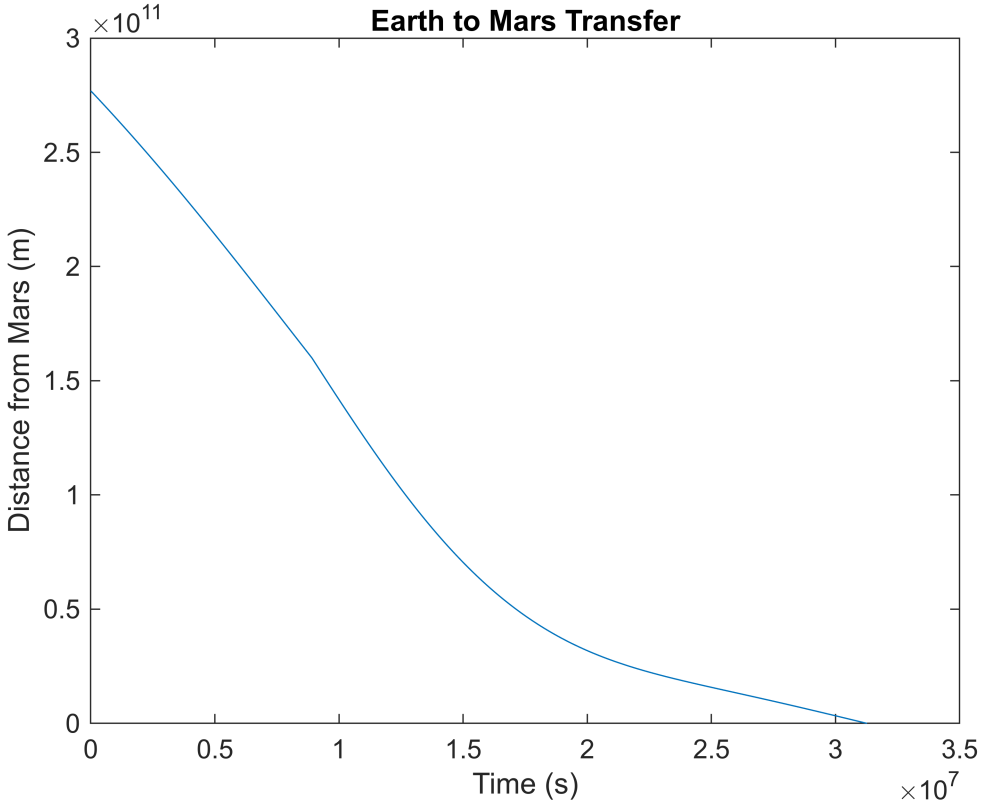
Constants and initial conditions are set in the Hohmann_setup.m script. It calls the hohmann_transfer.m function to compute the first transfer, finds the new angles of the planets after the first transfer, and runs hohmann_transfer.m again for the return trip.

```
addpath('Classes\','Functions\','Simulations\');
run("Hohmann_setup.m")
```

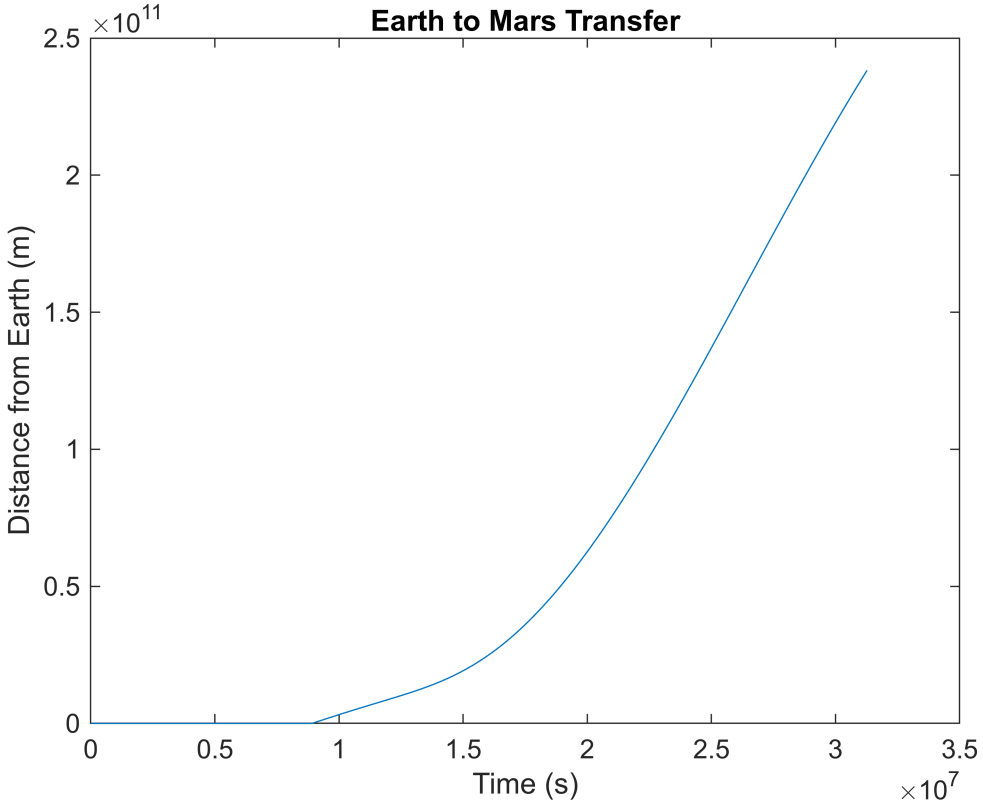
Verification

To make sure that the spacecraft stays at Earth before the start of the burn and reaches Mars by the end of the transfer, run the simulation Hohmann_outbound_sim.slx.

```
simOut = sim("Hohmann_outbound_sim.slx", 'ReturnWorkspaceOutputs', 'on');

plot_signal(simOut,"mars_dist","Distance from Mars (m)","Earth to Mars Transfer")
```

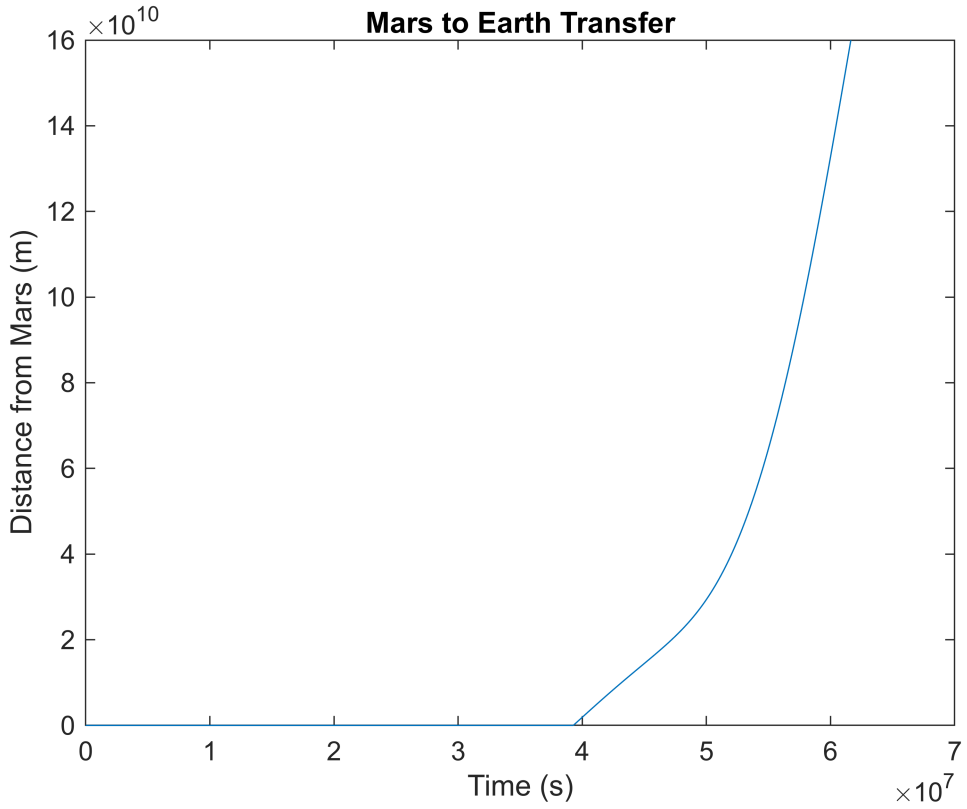



```
plot_signal(simOut,"earth_dist","Distance from Earth (m)","Earth to Mars Transfer")
```

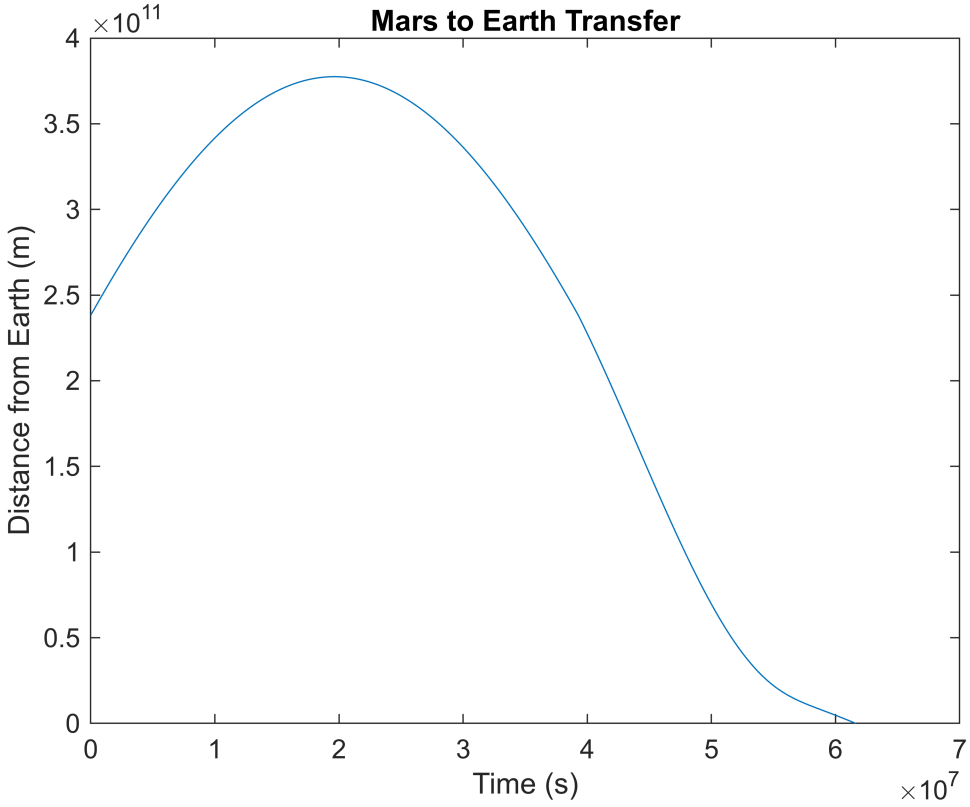


The return trip can also be verified with a nearly identical simulation: Hohmann_inbound_sim.slx. Note that t=0 for this new simulation is the end of the previous simulation.

```
simOut = sim("Hohmann_inbound_sim.slx", 'ReturnWorkspaceOutputs', 'on');

plot_signal(simOut,"mars_dist","Distance from Mars (m)","Mars to Earth Transfer")
```




```
plot_signal(simOut,"earth_dist","Distance from Earth (m)","Mars to Earth Transfer")
```



Launch to and Escape from Earth Orbit

Introduction

Now that the transfer from Earth to Mars has been calculated, we can go back to the beginning of the mission and calculate the maneuvers required to get a spacecraft from the launch site, into Earth orbit, and onto an escape trajectory to Mars.

Simplifying Assumptions

Since rocket launches are very complex situations that bring up many problems that don't directly relate to orbital mechanics, this project assumes the spacecraft is sitting on a platform at the desired orbital altitude that rotates with the Earth. In this case, "launching" to orbit means applying an impulse maneuver at the right time and direction to immediately end up in the target orbit. Also, assume that Earth's gravity and thrust are the only forces acting on the orbiter after launch.

Problem Statement

Given a launch site's latitude and longitude coordinates, calculate the impulse maneuver in the Earth-centered Equatorial coordinates required to put the orbiter into a circular parking orbit. This parking orbit should be in the plane of the ecliptic (i.e., inclination = 0 when using Earth-centered Ecliptic coordinates) to match with the calculations done for the Earth to Mars transfer. Then calculate the escape maneuver in the Earth-centered Equatorial coordinates required to put the spacecraft onto the Earth to Mars transfer.

Solution

Constants and initial conditions are set in the Earth_setup.m script. It calls the launch2orbit.m function to compute the launch maneuver and uses this data to call the escape.m funciton to find the escape manuver.

```
run("Earth_setup.m")
```

Verification

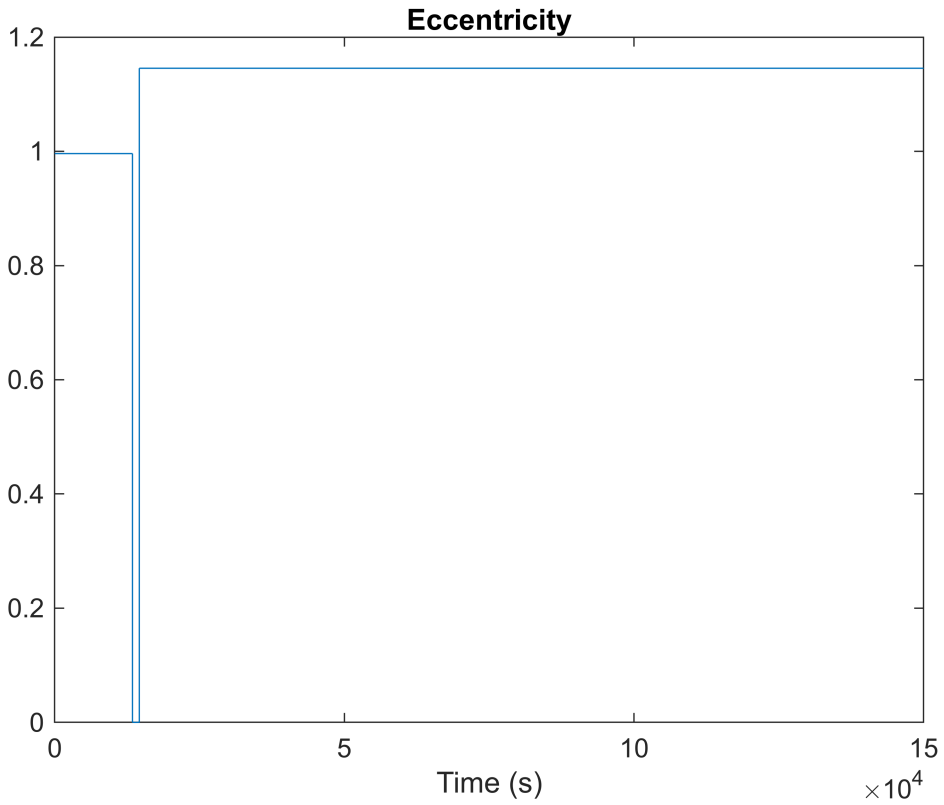
Both of the maneuvers in this section are simulated by the Earth_sim.slx simulation.

While the orbiter is sitting on its platform at orbital height, its eccentricity is near one. This is because its velocity is only due to the rotation of Earth before the launch. Additionally, the inclination will change as the plane of the "orbit" moves due to the Earth's rotation.

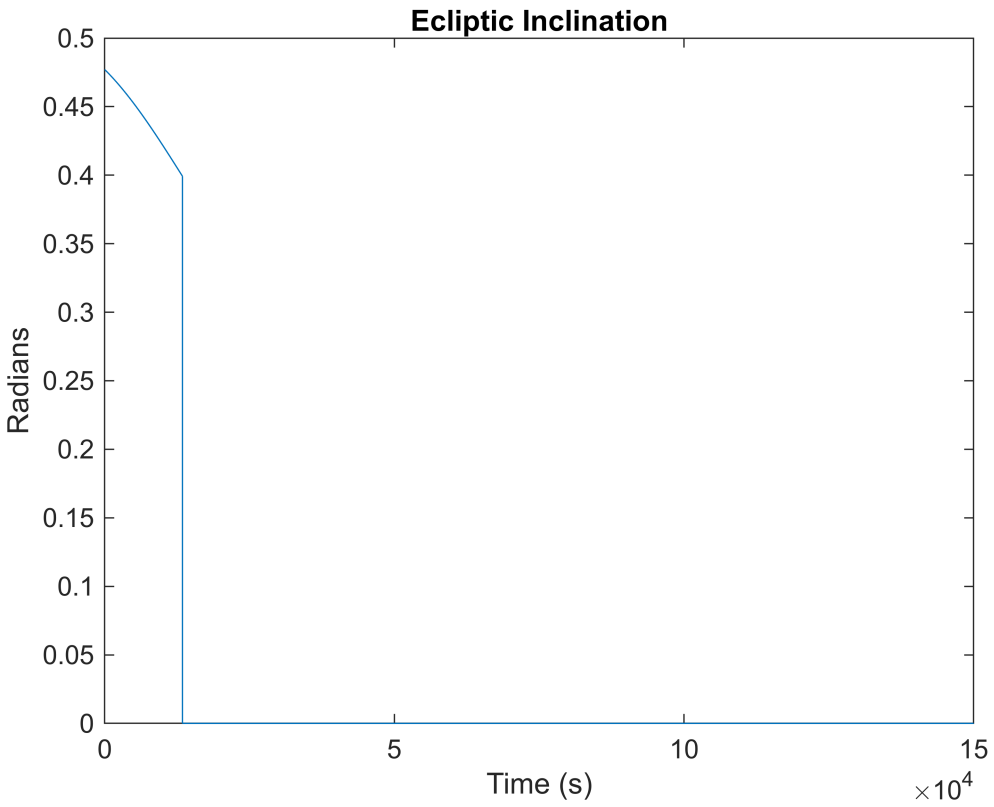
After the launch maneuver, the eccentricity and ecliptic inclination should be near zero since we want a circular orbit in the plane of the ecliptic.

Finally, once the orbiter performs the escape manuver, the eccentricity should be greater than 1 (it should be a hyperbolic escape orbit), the ecliptic eccentricity should stay near zero, the cosine of the angle between the actual and target escape velocities should be near 1, and the difference in magnitudes between these velocities should go to zero as time goes to infinity. Note that the velocity difference goes to zero very slowly. Running the simulation for longer shows better convergence.

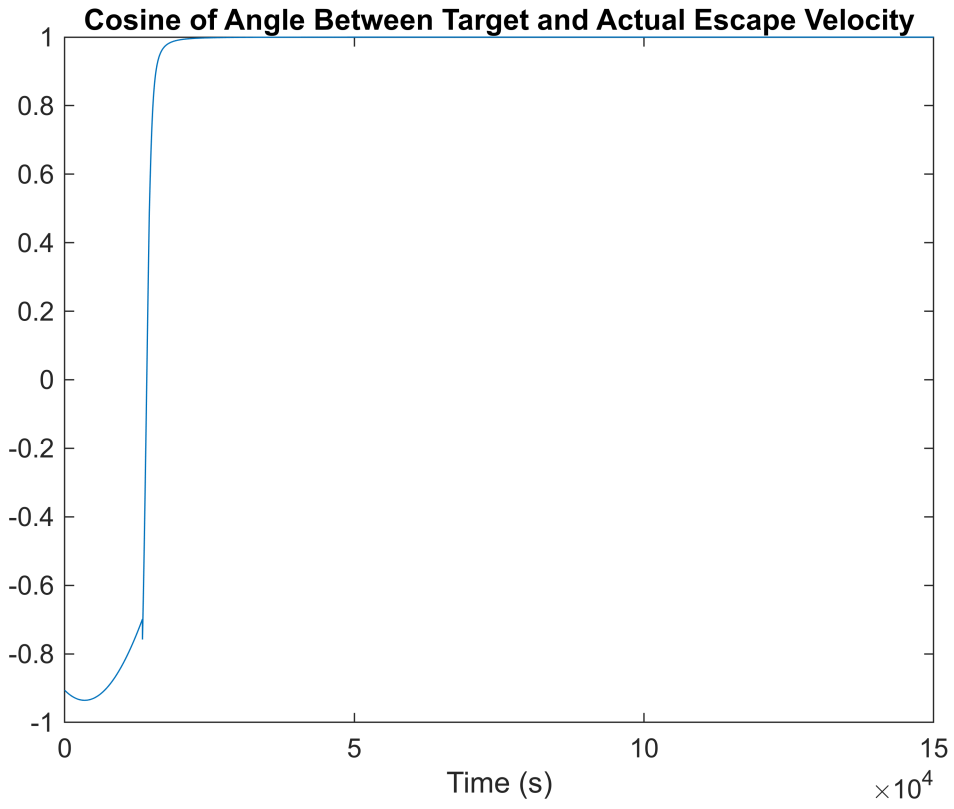
```
simOut = sim("Earth_sim.slx", 'ReturnWorkspaceOutputs', 'on');

plot_signal(simOut,"<e>","","Eccentricity")
```



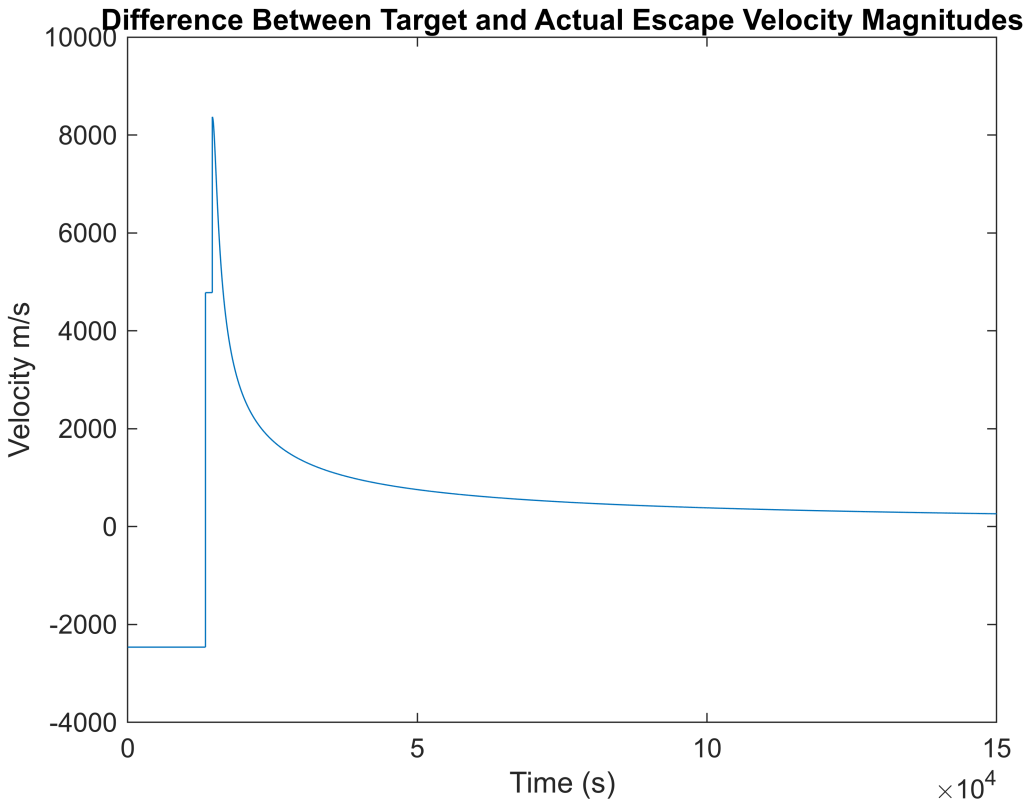

```
plot_signal(simOut,"<i>","Radians","Ecliptic Inclination")
```




```
plot_signal(simOut,"cos_theta","","Cosine of Angle Between Target and Actual Escape
Velocity")
```



```
plot_signal(simOut,"vel_mag_diff","Velocity m/s","Difference Between Target and
Actual Escape Velocity Magnitudes")
```



Mars Capture

Introduction

Now that the maneuvers around Earth and the interplanetary transfers to and from Mars have been calculated, it's time to find the maneuver that puts the orbiter into orbit around Mars. Not only that, but we need to make sure that the orbiter is put into the right plane to rendezvous with the sample canister waiting in Mars orbit.

Simplifying Assumptions

Mars' gravity only begins to affect the orbiter once it gets within one million km. Also, assume that the canister is waiting in a circular orbit. Lastly, it can be assumed that the orbiter begins on the asymptote formed by its hyperbolic trajectory with velocity equal to its encounter velocity.

Problem Statement

The orbiter will approach Mars with some initial velocity relative to the planet found in the interplanetary transfer step. The line passing through the center of Mars that is parallel to this velocity vector will be referred to as the line of approach. The initial distance from the line of approach will determine the periapsis and should be calculated to match that of the sample canister. This gives a circle of possible starting points for the orbiter at a given distance from Mars. The goal will be to choose the correct starting point along this circle such that the orbiter is in the plane of the sample canister. In reality, the exact position of the spacecraft would be determined by the specifics of the interplanetary transfer, but for the purposes of this project, it's much easier to assume

that it can be chosen at will along this predefined circle. Also, note that the orbiter's RA is fixed by the approach direction, meaning that the sample canister will have to be launched into a certain RA to meet the orbiter. Lastly, the orbiter will need to calculate the maneuver required to capture it into a circular Mars orbit given the initial conditions described above.

Solution

Constants and initial conditions are set in the Mars_capture_setup.m script. It calls the capture.m function to compute all the initial conditions and the Mars capture maneuver.

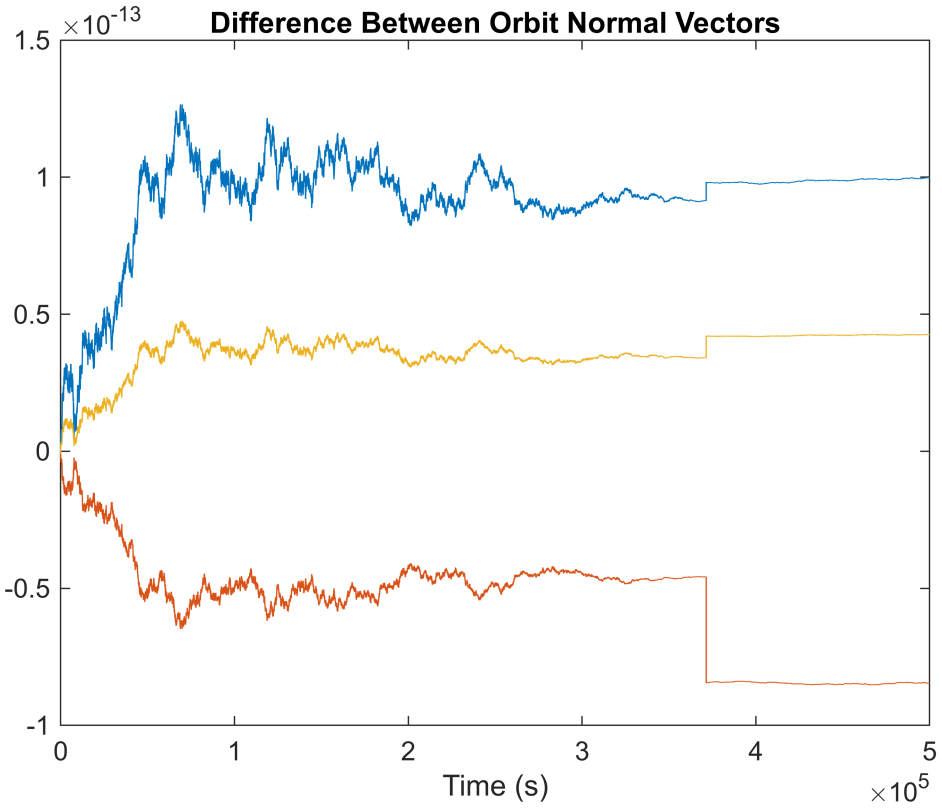
```
run("Mars_capture_setup.m")
```

Verification

This section is verified by the Mars_capture_sim.slx simulation.

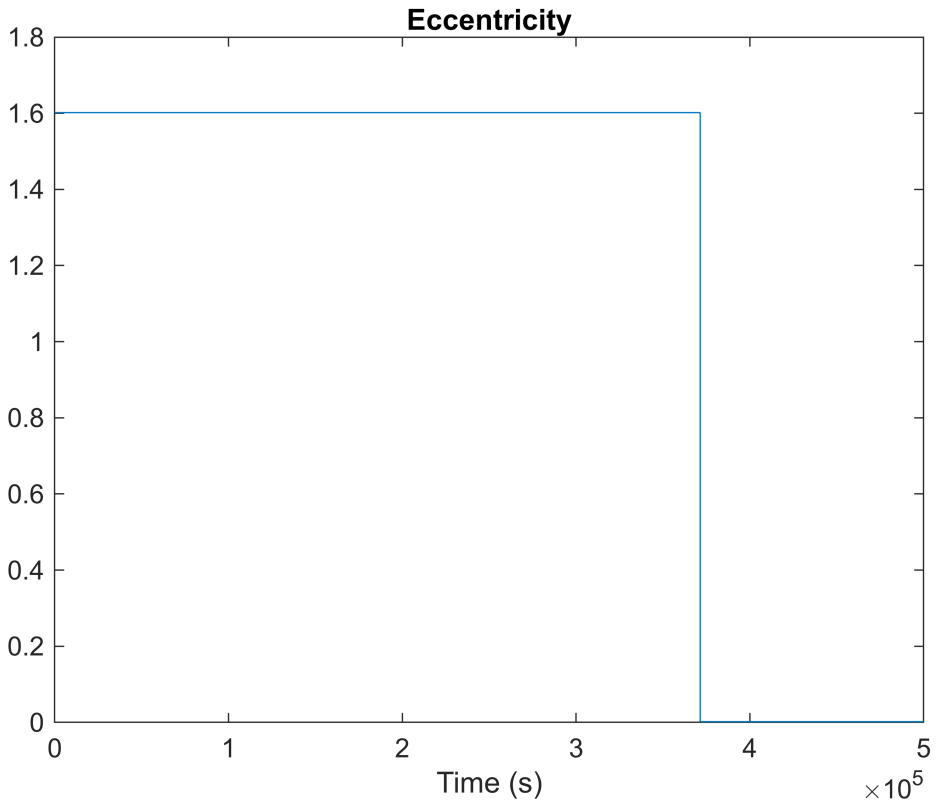
First of all, we need to make sure that the orbiter stays in the plane of the sample canister. This can be checked by looking at the difference in normal vectors of the planes. Of course, if they're in the same plane, it should be very near zero.

```
simOut = sim("Mars_capture_sim.slx", 'ReturnWorkspaceOutputs', 'on');

plot_signal(simOut,"plane_diff","","Difference Between Orbit Normal Vectors")
```



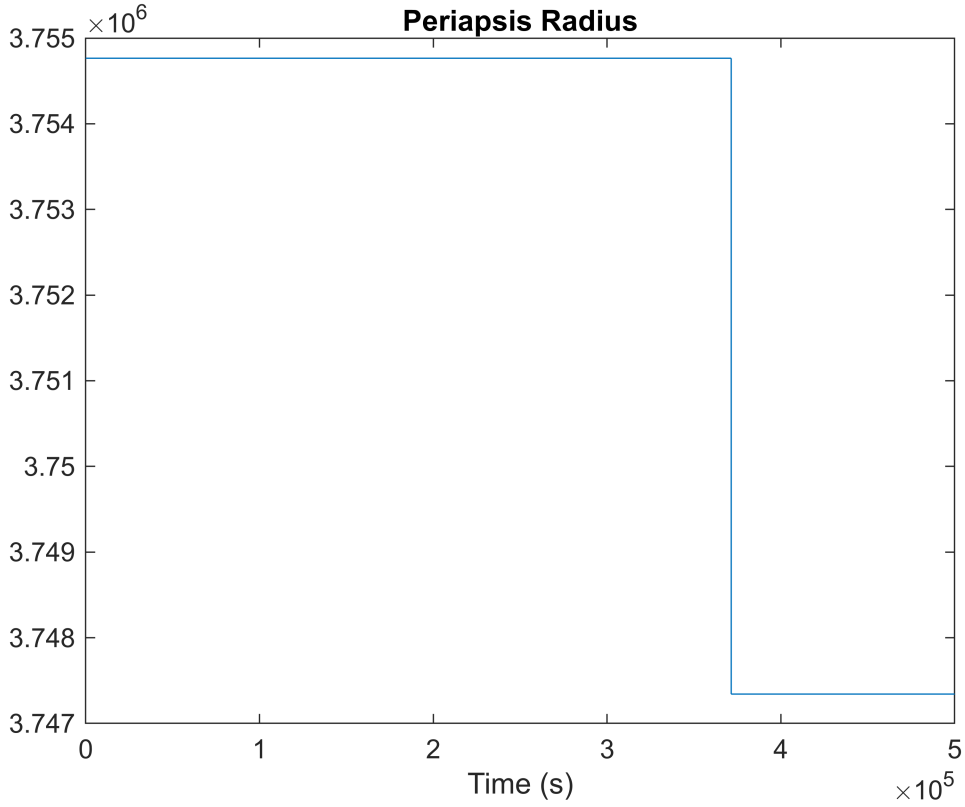
Additionally, we need to make sure that the orbiter's capture maneuver puts it into a circular orbit around Mars. This can be checked by looking at the eccentricity. Before the capture, it should be above one, and after the capture, it should be near zero.

```
plot_signal(simOut,"<e>","","Eccentricity")
```



Finally, check that the periapsis radius is nearly constant throughout the simulation and that it is close to the canister altitude of 3746 km.

```
plot_signal(simOut,"<r_p>","","Periapsis Radius")
```



Sample Canister Rendezvous

Introduction

Now that both the orbiter and the sample canister are in very similar orbits around Mars, it's time to calculate the two-impulse rendezvous maneuver that brings the two spacecraft into close proximity with zero relative velocity. As the name implies, there are two impulse maneuvers, one to start moving the orbiter towards the canister and another to stop it once it gets there.

Simplifying Assumptions

Assume that both spacecraft are already relatively close prior to the start of the rendezvous maneuvers. This allows the relative dynamics of the spacecraft to be modeled with the linearized equations of motion.

Problem Statement

Given the initial orbital elements of the two spacecraft, find the two impulse rendezvous maneuvers in Mars-centered Ecliptic coordinates. Note that, in theory, this problem can be solved for any rendezvous time, but in practice, it works better for shorter rendezvous times. That said, shorter transfer times require more delta-V to perform. Therefore, this trade-off should be taken into consideration when choosing a rendezvous time.

Solution

Constants and initial conditions are set in the Rendezvous_setup.m script. It calls the rendezvous.m function to compute the rendezvous maneuvers.

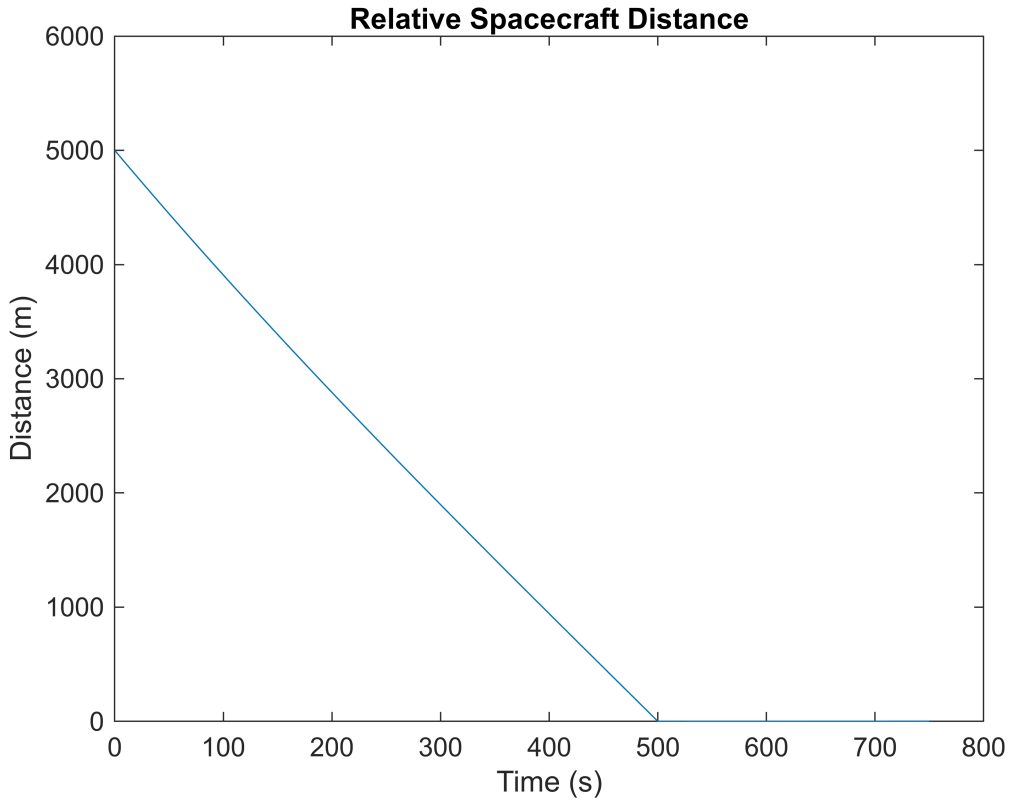
```
run("Rendezvous_setup.m")
```

Verification

This section is verified by the Rendezvous_sim.slx simulation.

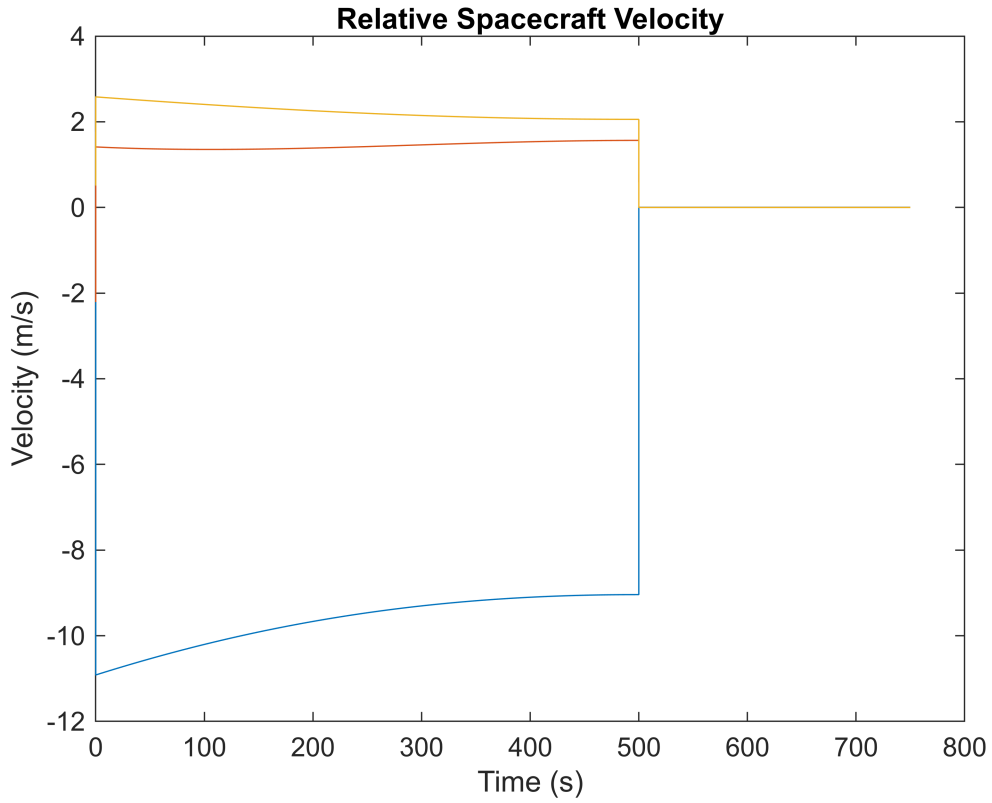
First of all, we need to make sure the orbiter reaches the canister. This can be checked by plotting the relative distance between the two over time.

```
simOut = sim("Rendezvous_sim.slx", 'ReturnWorkspaceOutputs', 'on');

plot_signal(simOut,"rel_dist","Distance (m)","Relative Spacecraft Distance")
```



Additionally, we need to make sure that the relative velocity between the two is negligible after both maneuvers are completed.

```
plot_signal(simOut,"rel_vel","Velocity (m/s)","Relative Spacecraft Velocity")
```



Mars Escape

Introduction

Now that the orbiter has rendezvoued with the sample canister, it's time to perform the escape burn that will put both of them onto the interplanetary trajectory towards Earth.

Problem Statement

Given the initial orbital elements of the orbiter, find the manuver in Mars-centered Ecliptic coordinates required to put it on the Mars to Earth transfer calculated earlier.

Solution

Constants and initial conditions are set in the Mars_escape_setup.m script. It calls the escape.m function to compute the escape maneuver.

```
run("Mars_escape_setup.m")
```

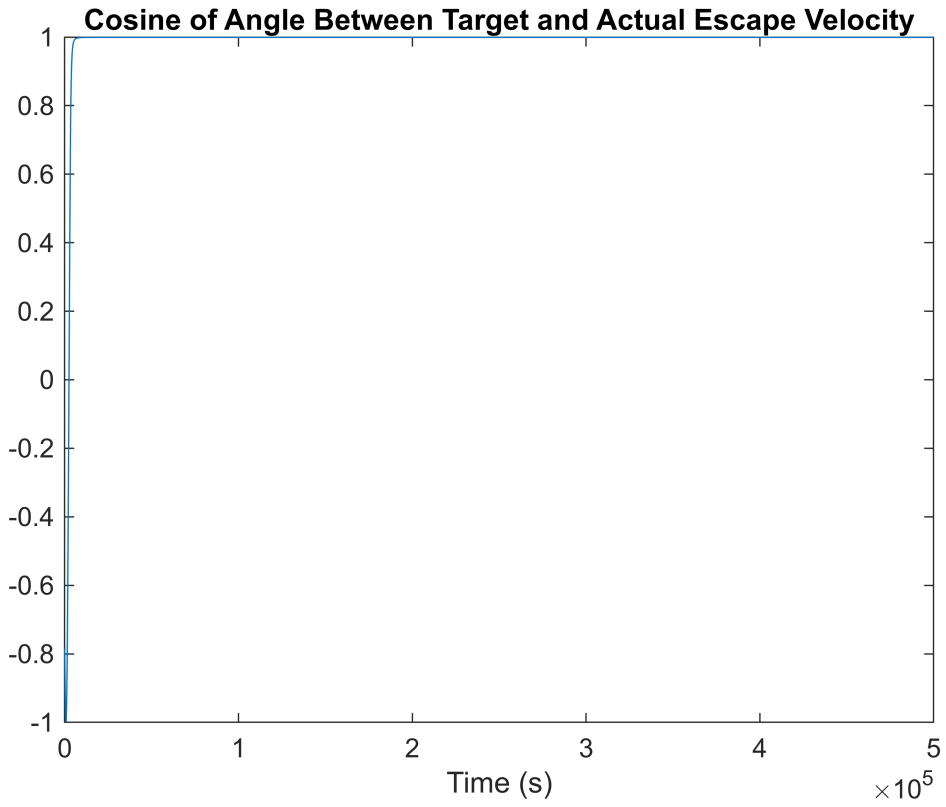
Verification

This section is verified by the Mars_escape_sim.slx simulation.

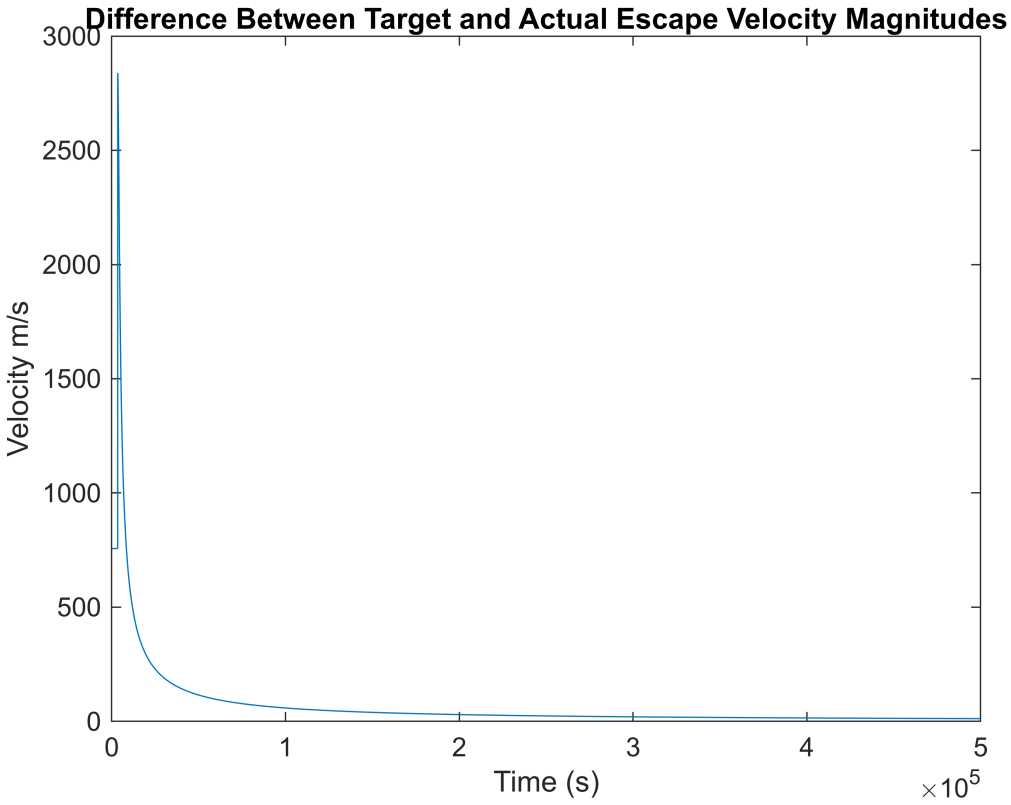
Once the orbiter performs the escape maneuver, the cosine of the angle between the actual and target escape velocities should be near 1, and the difference in magnitudes between these velocities should go to zero as time goes to infinity. Note that the velocity difference goes to zero very slowly. Running the simulation for longer shows better convergence to zero.

```
simOut = sim("Simulations/Mars_escape_sim.slx", 'ReturnWorkspaceOutputs', 'on');
```

```
plot_signal(simOut,"cos_theta","","Cosine of Angle Between Target and Actual Escape
Velocity")
```




```
plot_signal(simOut,"vel_mag_diff","Velocity m/s","Difference Between Target and
Actual Escape Velocity Magnitudes")
```


Difference Between Target and Actual Escape Velocity Magnitudes
Velocity m/s
Time (s)
3000
2500
2000
1500
1000
500
0
0
1
2
3
4
5
×10^5