

Orbital Maneuvers

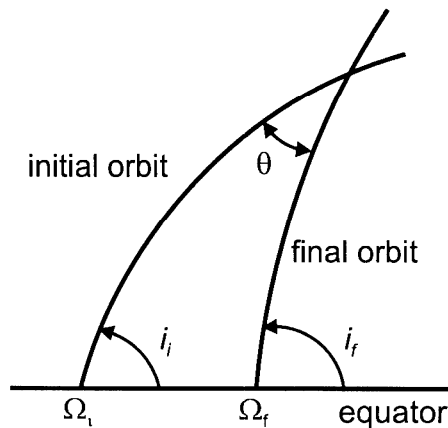
This document describes several MATLAB scripts that can be used to determine the characteristics of impulsive and finite burn maneuvers that modify orbits. The first script calculates the impulsive ΔV required to change a circular orbit, the second script determines the single impulse required to maneuver between any two intersecting orbits, and a third MATLAB script models a finite burn orbit transfer between two coplanar elliptical orbits. The impulsive ΔV assumption means that the velocity, but not the position, of the space vehicle is changed instantaneously.

This software suite also includes scripts that determine the characteristics of low-thrust orbital transfer, the impulsive maneuvers required to de-orbit from circular and elliptic orbits, and a script that calculates the characteristics of coplanar aero-assisted orbital transfer. A script to solve both the coplanar and non-coplanar Hohmann orbit transfer is also included along with scripts that perform a primer vector analysis of two impulse, coplanar Hohmann transfers and phasing or rendezvous maneuvers.

maneuver1.m – circular orbit plane change

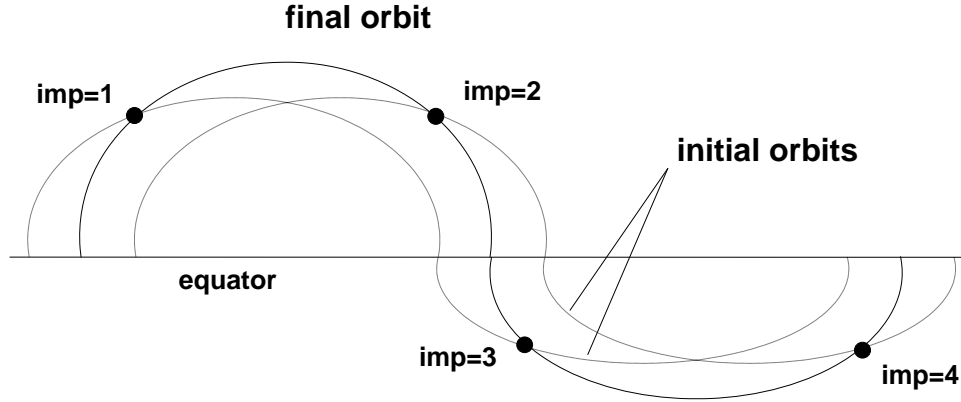
This section describes the geometry and equations associated with single impulse maneuvers that modify the inclination and/or right ascension of the ascending node (RAAN) of circular orbits.

The following diagram illustrates the geometry of this type of maneuver.



In this picture the orbital inclinations of the initial and final orbits are i_i and i_f , respectively. The RAAN of the initial orbit is Ω_i and Ω_f is the RAAN of the final orbit. The right ascension of the ascending node of an orbit is measured from the inertial x-axis along the equator in the direction of the Earth's rotation. From spherical trigonometry relationships θ is the angle between the two orbit planes.

The next diagram illustrates the possible points of intersection. From this ground track schematic we can see that there are two pairs of orbit intersections on both the initial and final orbits which depend on the relative RAAN between these two orbits.



The total plane change angle due to the modification of inclination and RAAN can be expressed as

$$\theta = \cos^{-1} \left[\sin i_i \sin i_f \cos(\Omega_f - \Omega_i) + \cos i_i \cos i_f \right]$$

We can define an index *imp* that depends on the *sign* of the RAAN change $\Delta\Omega = \Omega_f - \Omega_i$ as follows:

If $\Delta\Omega > 0$ then *imp* = 1 and 3

or

If $\Delta\Omega < 0$ then *imp* = 2 and 4.

It is convenient to define the location of impulses by their argument of latitude. The argument of latitude is the angle from the ascending node, measured along the orbital plane, to the point of interest. The argument of latitude is equal to the sum of the argument of perigee and true anomaly. Since for circular orbits there is no argument of perigee, the argument of latitude and true anomaly are identical.

The two possible arguments of latitude on the initial orbit depend on the values of *imp* as follows:

$$u_i = \text{integer}(imp/2)\pi - (-1)^{imp} u$$

where *u* is the impulse argument of latitude on the initial orbit given by

$$u = \cos^{-1} \left(\frac{\cos i_i \sin i_f \cos \Delta\Omega}{\sin i_i \sin \theta} \right)$$

We can determine the argument of latitude of an impulse on the final orbit by forming the unit position vectors from the ascending node to the impulse. The argument of latitude of the first opportunity on the final orbit is given by

Orbital Mechanics with MATLAB

$$u = \cos^{-1}(\mathbf{U}_1 \cdot \mathbf{U}_2)$$

where \mathbf{U}_1 is the unit position vector of the impulse on the initial orbit and \mathbf{U}_2 is the unit position vector to the ascending node of the final orbit. The argument of latitude of the second impulse opportunity on the final orbit is equal to 180 degrees plus this value.

The maneuver ΔV vector is given by the vector difference between the velocity vectors of the initial and final orbits as follows:

$$\Delta \mathbf{V} = \mathbf{V}_f - \mathbf{V}_i$$

These velocity vectors are evaluated at the points of orbital intersection. The scalar magnitude of the ΔV is determined from the components of this vector according to

$$\Delta V = \sqrt{\Delta V_x^2 + \Delta V_y^2 + \Delta V_z^2}$$

For the case where there is no RAAN change, the two impulse locations occur at the common ascending and descending nodes of both the initial and final orbits. The arguments of latitude of these two orbital points are 0 and 180 degrees, respectively.

The required ΔV can also be determined using vector manipulation. Unit vectors normal to each orbit plane can be defined as follows:

$$\mathbf{n}_i = \begin{bmatrix} \sin i_i \sin \Omega_i \\ -\sin i_i \cos \Omega_i \\ \cos i_i \end{bmatrix}$$

$$\mathbf{n}_f = \begin{bmatrix} \sin i_f \sin \Omega_f \\ -\sin i_f \cos \Omega_f \\ \cos i_f \end{bmatrix}$$

A unit vector along the intersection of the initial and final orbit planes is given by

$$\mathbf{m} = \frac{\mathbf{n}_f \times \mathbf{n}_i}{|\mathbf{n}_f \times \mathbf{n}_i|}$$

The velocity vector prior to the maneuver is calculated from

$$\mathbf{V}_i = \frac{\mathbf{n}_i \times \mathbf{m}}{|\mathbf{n}_i \times \mathbf{m}|} V_{lc}$$

The velocity vector after the maneuver is given by

Orbital Mechanics with MATLAB

$$\mathbf{V}_f = \frac{\mathbf{n}_f \times \mathbf{m}}{|\mathbf{n}_f \times \mathbf{m}|} V_{lc}$$

where V_{lc} is the local circular velocity at the maneuver altitude.

Finally, the maneuver ΔV vector is determined from

$$\Delta \mathbf{V} = \mathbf{V}_f - \mathbf{V}_i$$

The equations described here have been implemented in an interactive MATLAB script called `maneuver1.m`. This script will prompt you for the altitude, inclination and RAAN of both the initial and final orbits. A typical user interaction with this script is as follows.

```
program maneuver1

< one impulse transfer between circular orbits >

initial orbit

please input the altitude (kilometers)
(altitude > 0)
? 185

please input the orbital inclination (degrees)
(0 <= inclination <= 180)
? 28.5

please input the right ascension of the ascending node (degrees)
(0 <= raan <= 360)
? 100

final orbit

please input the orbital inclination (degrees)
(0 <= inclination <= 180)
? 45

please input the right ascension of the ascending node (degrees)
(0 <= raan <= 360)
? 120
```

The following is a typical screen display created with this script. For this example, the altitude of the orbits is 185 kilometers, the inclination and RAAN of the initial orbit were 28.5 and 100 degrees, respectively, and the inclination and RAAN of the final orbit were 45 and 120 degrees, respectively.

```
program maneuver1

< one impulse transfer between circular orbits >
```

Orbital Mechanics with MATLAB

```
solution # 1

initial orbit true anomaly      44.4493 degrees
final orbit true anomaly        28.2000 degrees

delta-V required                2733.7882 meters/second

solution # 2

initial orbit true anomaly      224.4493 degrees
final orbit true anomaly        208.2000 degrees

delta-V required                2733.7882 meters/second
```

maneuver2.m – single impulse transfer between intersecting orbits

This MATLAB script can be used to determine the single impulse propulsive maneuver between any two orbits that physically intersect. The orbital intersections are determined numerically and the ΔV evaluated using the vector methods described in the previous section.

The following MATLAB code, which is part of a support function called `intrsect.m`, searches for closest approach conditions. The software cycles through combinations of true anomaly on both the initial and final orbits looking for intersections with the `fminsearch` multi-dimensional optimization algorithm supplied with MATLAB. The scalar distance between the two orbits is calculated in a support function called `ca2sfun3.m`.

```
nf1 = 0;

for i = 0:1:11

    xg(1) = 30 * i * dtr;

    for j = 0:1:3

        xg(2) = 90 * j * dtr;

        % find minimum separation distance
        % true anomalies

        x = fminsearch('ca2sfun3', xg);

        rdelta = ca2sfun3(x);

        % check for valid solution

        if (rdelta < 1.0e-4)
            nf1 = nf1 + 1;
            x1f1(nf1) = mod(x(1), 2.0 * pi);
            x2f1(nf1) = mod(x(2), 2.0 * pi);
        end
    end
end
end
```

Orbital Mechanics with MATLAB

The check for intersection is performed in the `if . . . end` section of code where the tolerance is equal to `0.0001`. This procedure may find duplicate solutions that are then eliminated by the `intrsect` routine.

The source code for the closest approach objective function `ca2sfun3.m` is as follows:

```
function y = ca2sfun3(x)

% closest approach between two satellites
% objective function

% required by intrsect.m

% Orbital Mechanics with Matlab

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global mu oev1 oev2

oev1(6) = x(1);

oev2(6) = x(2);

[r1, v1] = orb2eci(mu, oev1);

[r2, v2] = orb2eci(mu, oev2);

% calculate separation distance

dr = r2 - r1;

y = norm(dr);
```

This function accepts the two-dimensional true anomaly vector defined by `x`, calculates the position and velocity vectors of both orbits at these points, and evaluates the scalar separation distance `y` between the initial and final orbits according to

$$\Delta \mathbf{r} = \mathbf{r}_i - \mathbf{r}_f$$

and

$$\Delta r = \sqrt{\Delta r_x^2 + \Delta r_y^2 + \Delta r_z^2}$$

Notice that the orbital elements of each orbit are passed in the global arrays `oev1` and `oev2` along with the gravitational constant `mu`.

This entire procedure has been implemented in a MATLAB script called `maneuver2.m` that will prompt you for the classical orbital elements, except true anomaly, of the initial and final orbits.

The following is a typical user interaction with this script.

Orbital Mechanics with MATLAB

program maneuver2

< one impulse transfer between intersecting orbits >

initial orbit

please input the semimajor axis (kilometers)

(semimajor axis > 0)

? **6678.4**

please input the orbital eccentricity (non-dimensional)

(0 <= eccentricity < 1)

? **.0075**

please input the orbital inclination (degrees)

(0 <= inclination <= 180)

? **28.5**

please input the argument of perigee (degrees)

(0 <= argument of perigee <= 360)

? **30**

please input the right ascension of the ascending node (degrees)

(0 <= raan <= 360)

? **0**

final orbit

please input the semimajor axis (kilometers)

(semimajor axis > 0)

? **18953.14**

please input the orbital eccentricity (non-dimensional)

(0 <= eccentricity < 1)

? **.6556**

please input the orbital inclination (degrees)

(0 <= inclination <= 180)

? **28.5**

please input the argument of perigee (degrees)

(0 <= argument of perigee <= 360)

? **300**

please input the right ascension of the ascending node (degrees)

(0 <= raan <= 360)

? **0**

The following is the screen display created with this script. Notice that the script also provides the local-vertical, local horizontal (LVLH) pitch and yaw angles for each impulse. These angles are measured with respect to the initial orbit. The orbital elements of the initial and final orbits for this example are as follows:

Orbital Mechanics with MATLAB

	initial orbit	final orbit	
a	6678.4	18953.14	kilometers
e	0.0075	0.6556	---
i	28.5	28.5	degrees
ω	30	300	degrees
Ω	0	0	degrees

For this example the orbital inclination and RAAN of the initial and final orbits are the same (since the orbits are coplanar) that explains why the yaw angle at both maneuvers is zero.

```
program maneuver2

< one impulse transfer between intersecting orbits >

solution # 1

orbit #1 true anomaly at intersection      288.3946 degrees
orbit #2 true anomaly at intersection      18.3946 degrees

delta-V at intersection                    2482.0201 meters/second
delta-V LVLH pitch angle                   31.8951 degrees
delta-V LVLH yaw angle                     0.0000 degrees

solution # 2

orbit #1 true anomaly at intersection      249.4843 degrees
orbit #2 true anomaly at intersection      339.4843 degrees

delta-V at intersection                    2489.0024 meters/second

delta-V LVLH pitch angle                   -32.6039 degrees
delta-V LVLH yaw angle                     0.0000 degrees
```

maneuver3.m – finite burn orbit transfer

This MATLAB script simulates a single finite burn maneuver for orbit transfer between two coplanar elliptical orbits. The user can choose from three types of thrust vector “steering” during the maneuver. This application assumes a constant thrust level and propellant flow rate. The single finite burn maneuver occurs at the mutual perigee of the initial and final orbits, and the burn is “centered” about perigee.

The following is a brief description about each type of steering method.

Tangential

For this type of steering the thrust pointing direction is tangential to the instantaneous radius vector to the spacecraft and in the direction of the orbital motion. This implies that both the pitch and yaw angles are always zero during the finite burn maneuver.

Gravity turn

For this type of steering the thrust pointing direction is always aligned with the instantaneous velocity vector of the vehicle.

Linear pitch angle

For this third type of steering, the pitch angle of the maneuver changes linearly during the burn. The angular range over which the maneuver occurs is given by

$$\alpha = 2\pi \left(\frac{t_d}{\tau} \right)$$

where τ is the orbital period of the initial orbit. The initial and final pitch angles of the maneuver are given by

$$\theta_i = -\frac{\alpha}{2}$$

and

$$\theta_f = +\frac{\alpha}{2}$$

The pitch angle at any time t is determined from the following expression

$$\theta(t) = \theta_i - \frac{d\theta}{dt} (t - t_{ign})$$

where the pitch rate is given by $\frac{d\theta}{dt} = (\theta_i - \theta_f) / t_d$ and t_{ign} is the ignition time of the maneuver.

In these expressions t_d is the thrust duration of the maneuver.

The thrust duration is calculated from the ideal rocket equation with the expression

$$t_d = \frac{g I_{sp} m_p}{T}$$

where I_{sp} is the specific impulse and m_p is the propellant mass required for the maneuver. The propellant mass is also calculated from the ideal rocket equation.

The “gravity loss” for a finite burn maneuver is given by:

$$\Delta V_g = \int_{t_{ign}}^{t_{bo}} g \sin \gamma dt$$

where g is the scalar acceleration of gravity at the location of the maneuver, γ is the instantaneous flight path angle, t_{ign} is the ignition time of the maneuver and t_{bo} is the burnout or termination time of the burn. The gravity loss occurs because the $g \sin \gamma$ term turns the vehicle away from the optimal thrust direction during a finite burn maneuver.

The inertial delta-velocity added to the vehicle by the finite burn is given by

$$\Delta \mathbf{V}_i = \int_{t_{ign}}^{t_{bo}} T \mathbf{u}_{ECI} dt$$

The total acceleration of the spacecraft is a combination of “Keplerian” gravity and thrust acceleration according to the following expression:

$$\mathbf{a} = \mathbf{g} + \left(\frac{T}{m} \right) \mathbf{u}_{ECI}$$

In this equation T is the thrust level, m is the instantaneous mass of the vehicle and \mathbf{u}_{ECI} is the inertial unit pointing vector along which the thrust is applied. This script numerically integrates this system of nonlinear vector differential equations using the `rkf78` function while accounting for the change in spacecraft mass due to propellant expenditure.

The transformation of a unit pointing vector in the radial-tangential-normal (RTN) coordinate system centered at the spacecraft \mathbf{u}_{RTN} to an Earth-centered-inertial (ECI) unit pointing vector \mathbf{u}_{ECI} is given by the following operation:

$$\mathbf{u}_{ECI} = \begin{bmatrix} -h_x & (\mathbf{h} \times \mathbf{r})_x & r_x \\ -h_y & (\mathbf{h} \times \mathbf{r})_y & r_y \\ -h_z & (\mathbf{h} \times \mathbf{r})_z & r_z \end{bmatrix} \mathbf{u}_{RTN}$$

In this matrix \mathbf{h} is the angular momentum vector and all x , y and z components of these matrix elements are unit vectors.

The RTN unit pointing vector at any mission time t is determined from the pitch and yaw angles as follows:

$$\mathbf{u}_{RTN}(t) = \begin{Bmatrix} \sin \varphi(t) \cos \theta(t) \\ \cos \varphi(t) \cos \theta(t) \\ \sin \theta(t) \end{Bmatrix}$$

For the coplanar case modeled in this MATLAB script, the yaw or out-of-plane angle φ is always zero.

The following is a typical user interaction with this script.

Orbital Mechanics with MATLAB

```
program maneuver3

< finite burn orbit transfer between elliptical orbits >

initial orbit

please input the perigee altitude (kilometers)
(perigee altitude > 0)
? 300

please input the apogee altitude (kilometers)
(apogee altitude >= perigee altitude)
? 300

final orbit

please input the apogee altitude (kilometers)
(apogee altitude >= perigee altitude)
? 500

please input the thrust (newtons)
(thrust > 0)
? 400

please input the specific impulse (seconds)
(specific impulse > 0)
? 300

please input the initial mass (kilograms)
(initial mass > 0)
? 4000

please select the steering method

<1> tangential

<2> gravity turn

<3> linear pitch rate

? 3
```

The following is the script output for this example.

```
program maneuver3

< finite burn orbit transfer between elliptical orbits >

initial orbit

perigee altitude          300.000000 kilometers
apogee altitude          300.000000 kilometers
```

Orbital Mechanics with MATLAB

initial mass 4000.000000 kilograms
thrust 400.000000 newtons
specific impulse 300.000000 seconds
exhaust velocity 2941.995000 meters/seconds
propellant flow rate 0.135962 kilograms/seconds

impulsive maneuver

delta-v 56.781639 meters/second
thrust duration 562.371931 seconds
propellant mass 76.461303 kilograms

finite burn maneuver - linear pitch rate steering

delta-v 56.781626 meters/second
gravity loss 5.778402 meters/second
propellant mass 76.461303 kilograms
pitch rate -0.066284 degrees/second

final orbit

perigee altitude 299.978878 kilometers
apogee altitude 496.473985 kilometers

sma (km)	eccentricity	inclination (deg)	argper (deg)
+6.77636273153397e+003	+1.44985676268658e-002	+0.00000000000000e+000	+5.17362827084665e-002
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
+0.00000000000000e+000	+1.87112067751264e+001	+1.87629430578349e+001	+9.25240641504207e+001

If this same maneuver was performed with tangential steering, the finite burn and final orbit results are as follows:

finite burn maneuver - tangential steering

delta-v 55.779011 meters/second
gravity loss 7.769443 meters/second
propellant mass 76.461247 kilograms

final orbit

Orbital Mechanics with MATLAB

```
perigee altitude      301.7276 kilometers
apogee altitude      498.2205 kilometers
```

If this same maneuver was performed with gravity turn steering, the results are as follows:

```
finite burn maneuver - gravity turn steering
delta-v              55.793099 meters/second
gravity loss         7.783571 meters/second
propellant mass     76.461247 kilograms

final orbit
perigee altitude    301.7154 kilometers
apogee altitude    498.2330 kilometers
```

ltot.m – low thrust orbit transfer

This MATLAB script determines the characteristics of low-thrust orbital transfer between two nonplanar circular orbits. The numerical method used in this script is described in Chapter 14 of the book *Orbital Mechanics* by V. Chobotov and the technical paper, “The Reformulation of Edelbaum's Low-thrust Transfer Problem Using Optimal Control Theory” by J. A. Kechichian, AIAA-92-4576-CP. The original Edelbaum algorithm is described in “Propulsion Requirements for Controllable Satellites”, ARS Journal, August 1961, pages 1079-1089. This algorithm is valid for total inclination changes Δi given by $0 < \Delta i < 114.6^\circ$. This algorithm assumes that the thrust acceleration magnitude and spacecraft mass are both constant during the orbit transfer.

The initial thrust vector yaw angle β_0 is given by the following expression

$$\tan \beta_0 = \frac{\sin\left(\frac{\pi}{2}\Delta i\right)}{\frac{V_0}{V_f} - \cos\left(\frac{\pi}{2}\Delta i\right)}$$

where the speed on the initial circular orbit is $V_0 = \sqrt{\mu/r_0}$ and the speed on the final circular orbit is $V_f = \sqrt{\mu/r_f}$. In these equations $r_0 = r_e + h_0$ is the geocentric radius of the initial orbit, $r_f = r_e + h_f$ is the geocentric radius of the final orbit, r_e is the radius of the Earth and μ is the gravitational constant of the Earth. The initial altitude is h_0 and the final altitude is h_f .

The total velocity change required for a low-thrust orbit transfer is given by

Orbital Mechanics with MATLAB

$$\Delta V = V_0 \cos \beta_0 - \frac{V_0 \sin \beta_0}{\tan\left(\frac{\pi}{2} \Delta i + \beta_0\right)}$$

The total transfer time is given by $t = \Delta V/f$ where f is the thrust acceleration. The time evolution of the yaw angle, speed and inclination change are given by the following three expressions:

$$\beta(t) = \tan^{-1}\left(\frac{V_0 \sin \beta_0}{V_0 \cos \beta_0 - ft}\right)$$
$$V(t) = \sqrt{V_0^2 - 2V_0 ft \cos \beta_0 + f^2 t^2}$$
$$\Delta i(t) = \frac{2}{\pi} \left[\tan^{-1}\left(\frac{ft - V_0 \cos \beta_0}{V_0 \sin \beta_0}\right) + \frac{\pi}{2} - \beta_0 \right]$$

This MATLAB script will prompt you for the initial and final altitudes and orbital inclinations, and the thrust acceleration. The following is a typical user interaction with this script. It illustrates an orbital transfer from a low Earth orbit (LEO) with an inclination of 28.5° to a geosynchronous Earth orbit (GSO) with an orbital inclination of 0° . The thrust acceleration for this example is $3.5\text{E-}4$ meters/second².

```
Low-thrust Orbit Transfer Analysis
```

```
please input the initial altitude (kilometers)
? 621.86
```

```
please input the final altitude (kilometers)
? 35787.86
```

```
please input the initial orbital inclination (degrees)
(0 <= inclination <= 180)
? 28.5
```

```
please input the final orbital inclination (degrees)
(0 <= inclination <= 180)
? 0
```

```
please input the thrust acceleration (m/sec^2)
? 3.5e-4
```

The following is the output created for this example.

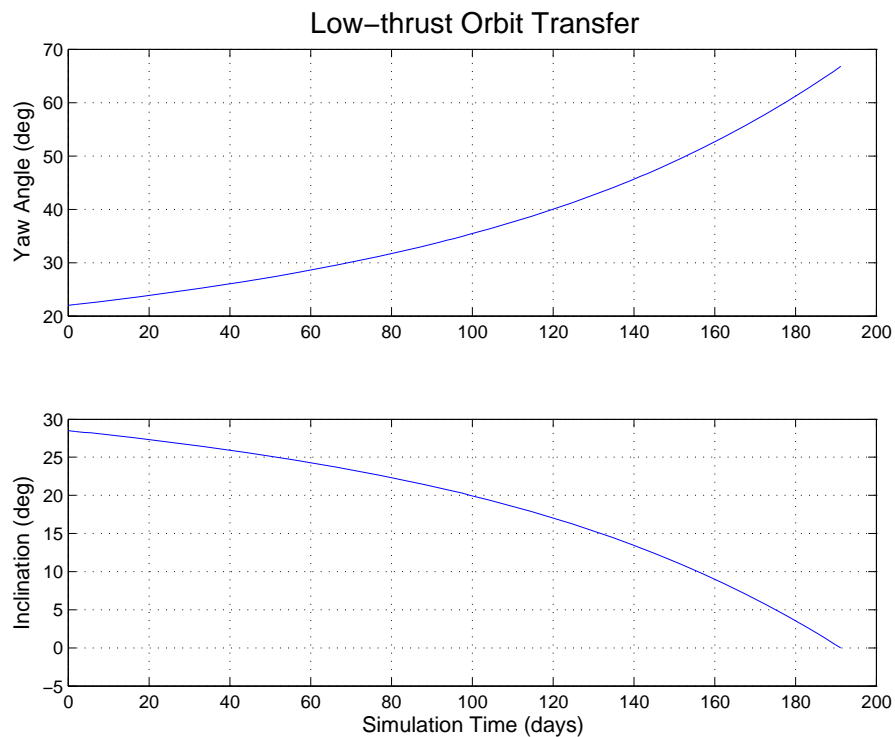
```
Low-thrust Orbit Transfer Analysis
```

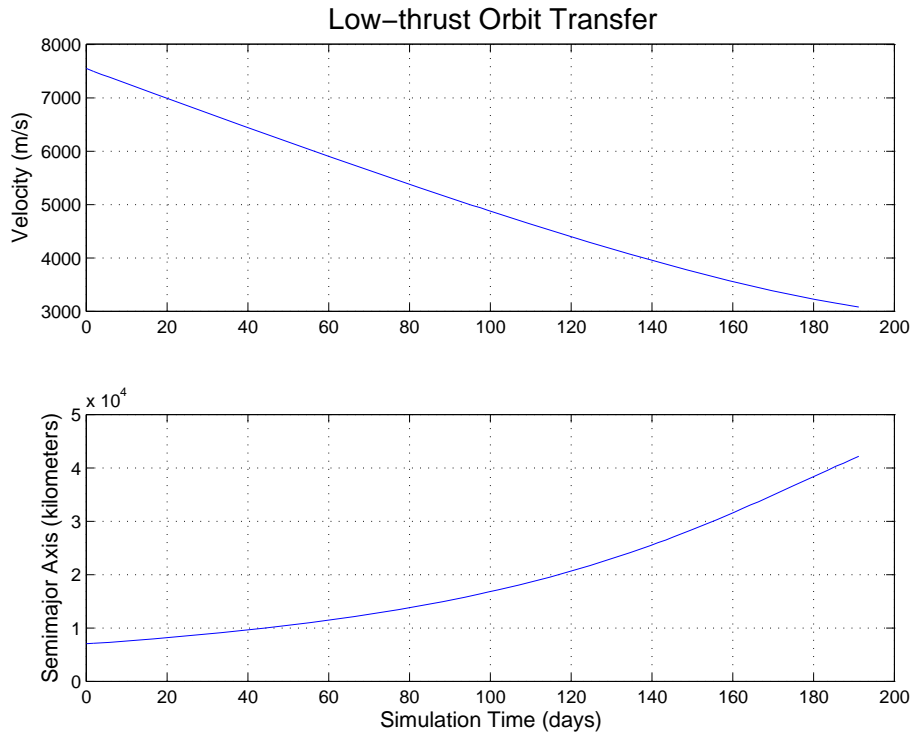
```
initial orbit altitude      621.8600 kilometers
```

Orbital Mechanics with MATLAB

initial orbit inclination	28.5000 degrees
initial orbit velocity	7546.0538 meters/second
final orbit altitude	35787.8600 kilometers
final orbit inclination	0.0000 degrees
final orbit velocity	3074.5936 meters/second
total inclination change	28.5000 degrees
total delta-v	5783.7751 meters/second
thrust duration	191.2624 days
initial yaw angle	21.9850 degrees
thrust acceleration	0.000350 meters/second ²

The software will also graphically display the time evolution of the thrust vector yaw angle, speed, inclination change and semimajor axis. The graphics for this example are as follows:





sep_ltot.m – low-thrust orbit transfer using solar-electric propulsion

This interactive MATLAB script can be used to determine the characteristics of continuous, low-thrust orbital transfer between two *non-coplanar* circular orbits using solar-electric propulsion (SEP). The numerical method used in this script is identical to the technique used in the `ltot.m` script described previously.

The propulsive thrust provided by an SEP system is given by

$$T = \frac{2\eta P}{gI_{sp}}$$

where η is the non-dimensional propulsive efficiency, P is the input power in kilowatts, g is the acceleration of gravity in meters/second and I_{sp} is the specific impulse in seconds. The quantity gI_{sp} is also called the exhaust velocity. Note that with these metric units the thrust will be in milli-newtons. The thrust acceleration required in the equations to follow is equal to $a_T = T/m$ where m is the mass of the spacecraft.

This MATLAB script will prompt you for the initial and final altitudes and orbital inclinations, and the SEP propulsive characteristics. The following is a typical user interaction with this script. It illustrates an orbital transfer from a low Earth orbit (LEO) with an inclination of 28.5° to a geosynchronous Earth orbit (GSO) with an orbital inclination of 0° .

Orbital Mechanics with MATLAB

SEP Low-thrust Orbit Transfer Analysis

please input the initial altitude (kilometers)
? **621.86**

please input the final altitude (kilometers)
? **35787.86**

please input the initial orbital inclination (degrees)
(0 <= inclination <= 180)
? **28.5**

please input the final orbital inclination (degrees)
(0 <= inclination <= 180)
? **0**

please input the initial spacecraft mass (kilograms)
? **1147.732571**

please input the SEP propulsive efficiency (non-dimensional)
? **.65**

please input the SEP input power (kilowatts)
? **10**

please input the SEP specific impulse (seconds)
? **3300**

The following is the output created for this example.

initial orbit altitude	621.8600 kilometers
initial orbit inclination	28.5000 degrees
initial orbit velocity	7546.0538 meters/second
final orbit altitude	35787.8600 kilometers
final orbit inclination	0.0000 degrees
final orbit velocity	3074.5936 meters/second
propulsive efficiency	0.6500
input power	10.0000 kilowatts
specific impulse	3300.0000 seconds
thrust	0.4017 newtons
initial spacecraft mass	1147.7326 kilograms

Orbital Mechanics with MATLAB

final spacecraft mass	959.8933 kilograms
propellant mass	187.8393 kilograms
total inclination change	28.5000 degrees
total delta-v	5783.7751 meters/second
thrust duration	191.2624 days
initial yaw angle	21.9850 degrees
thrust acceleration	0.000350 meters/second^2

The software will also graphically display the time evolution of the thrust vector yaw angle, spacecraft speed, inclination change and semimajor axis.

cdeorbit.m – single impulse de-orbit from a circular orbit

This MATLAB script calculates the single impulsive maneuver required to establish a reentry altitude and flight path angle relative to an initial circular orbit. The algorithm uses a tangential ΔV applied opposite to the velocity vector to establish the de-orbit trajectory.

The scalar magnitude of the de-orbit maneuver is determined from the following expression:

$$\Delta V = V_{ce} \sqrt{\frac{1}{\tilde{r}}} \left\{ 1 - \frac{\sqrt{2(\tilde{r}-1)}}{\sqrt{\left(\frac{\tilde{r}}{\cos \gamma_e}\right)^2 - 1}} \right\}$$

where

$$\tilde{r} = \frac{h_i + r_{eq}}{h_e + r_{eq}} = \text{radius ratio}$$

$$V_{ce} = \sqrt{\frac{\mu}{(h_e + r_{eq})}} = \text{local circular velocity at reentry altitude}$$

γ_e = reentry flight path angle

h_i = altitude of initial circular orbit

h_e = reentry altitude

r_{eq} = Earth equatorial radius

μ = Earth gravitational constant

This algorithm is described in “Deboost from Circular Orbits”, A. H. Milstead, *The Journal of the Astronautical Sciences*, Vol. XIII, No. 4, pages 170-171, Jul-Aug., 1966.

Orbital Mechanics with MATLAB

The following is a typical user interaction with this script.

```
program cdeorbit  
  
< single impulse deorbit from circular orbits >  
  
please input the initial altitude (kilometers)  
? 1000  
  
please input the entry altitude (kilometers)  
? 100  
  
please input the entry flight path angle (degrees)  
? -2
```

The following is the script output created for this example.

```
program cdeorbit  
  
< single impulse deorbit from circular orbits >  
  
initial altitude      1000.000000 kilometers  
entry altitude       100.000000 kilometers  
entry fpa            -2.000000 degrees  
  
entry trajectory  
semimajor axis       6896.07935765 kilometers  
eccentricity         0.06990358  
argument of perigee  180.00000000 degrees  
perigee altitude     35.87871531 kilometers  
apogee altitude      1000.00000000 kilometers  
entry true anomaly   328.04948058 degrees  
entry velocity       8078.31275892 meters/second  
impulse-to-entry time 40.13350666 minutes  
deorbit delta-v      261.55416617 meters/second
```

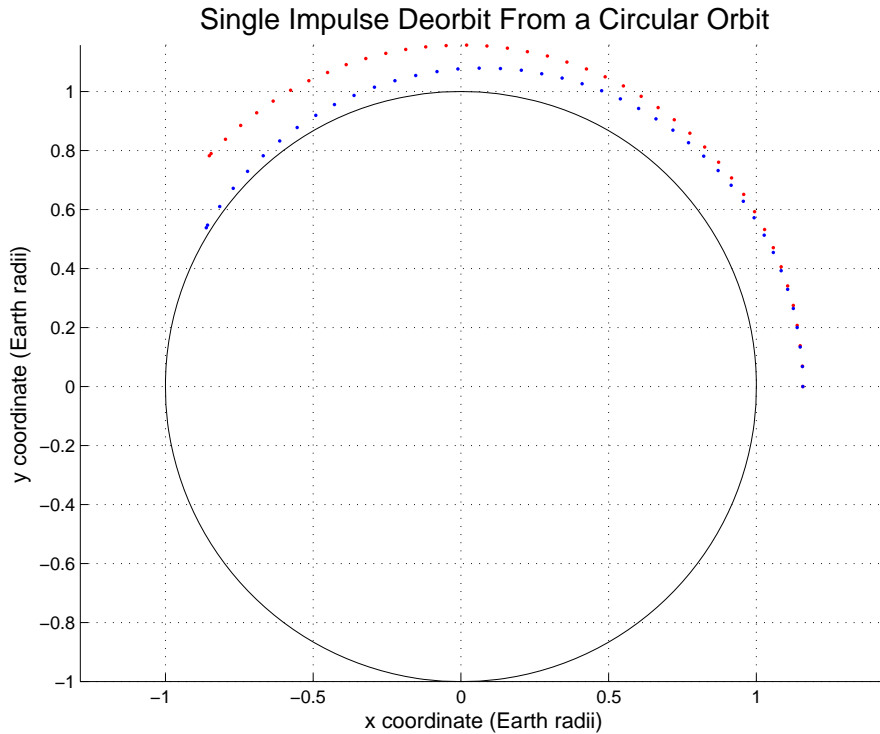
The software will also calculate and display the entry velocity and flight path angle relative to a rotating Earth. The following is the relative flight information for this example.

```
relative flight path coordinates
```

Orbital Mechanics with MATLAB

flight path angle	-2.12418719 degrees
velocity magnitude	7.60622497 kilometers/second

The software will also graphically display the initial circular orbit and the de-orbit trajectory. The graphic display for this example is as follows where the red dots represent the original circular orbit and the blue dots represent the de-orbit trajectory, both at one minute intervals.



The maneuver creates an elliptical de-orbit trajectory with an apogee located at the maneuver point. The apogee altitude of this trajectory is equal to the altitude of the initial circular orbit.

edeorbit.m – single impulse de-orbit from an elliptical orbit

This MATLAB script calculates the single impulsive maneuver required to establish a reentry altitude and flight path angle relative to an initial elliptical orbit. The algorithm uses a tangential ΔV applied opposite to the velocity vector at apogee of the initial orbit to establish the de-orbit trajectory that enters the Earth's atmosphere.

The scalar magnitude of this de-orbit delta-v is given by

$$\Delta V = \sqrt{\frac{\mu}{r_e}} \left(\sqrt{\frac{2\tilde{r}_p}{\tilde{r}_a(\tilde{r}_a + \tilde{r}_p)}} - \sqrt{\frac{2(\tilde{r}_a - 1)}{\tilde{r}_a(\tilde{r}_a^2 - \cos^2 \gamma_e)}} \cos \gamma_e \right)$$

where

r_e = geocentric radius at the entry altitude

$\tilde{r}_a = r_a / r_e$

$\tilde{r}_p = r_p / r_e$

γ_e = flight path angle at entry

r_a = apogee radius of the initial elliptical orbit

r_p = perigee radius of the initial elliptical orbit

μ = gravitational constant of the Earth

The true anomaly at entry can be determined from the following series of equations:

$$\sin \theta_e = \frac{\dot{r}}{e_d} \sqrt{\frac{a_d (1 - e_d^2)}{\mu}}$$

$$\cos \theta_e = \frac{a_d (1 - e_d^2)}{e_d r_e} - \frac{1}{e_d}$$

$$\theta_e = \tan^{-1}(\sin \theta_e, \cos \theta_e)$$

where

e_d = eccentricity of deorbit trajectory

a_d = semimajor axis of deorbit trajectory

$$\dot{r} = -\sqrt{\frac{\mu [2a_d r_e - r_e^2 - a_d^2 (1 - e_d^2)]}{a_d r_e^2}}$$

and the inverse tangent is a four quadrant operation.

The time of flight between perigee and the entry true anomaly θ_e is given by:

$$t(\theta_e) = \frac{\tau}{2\pi} \left[2 \tan^{-1} \left\{ \sqrt{\frac{1 - e_d}{1 + e_d}} \tan \frac{\theta_e}{2} \right\} - \frac{e_d \sqrt{1 - e_d^2} \sin \theta_e}{1 + e_d \cos \theta_e} \right]$$

In this equation τ is the orbital period of the de-orbit trajectory.

Therefore, the flight time between the de-orbit impulse time and entry is given by

$$\Delta t = t(\theta_e) - t(180^\circ) = t(\theta_e) - \frac{\tau}{2}$$

Orbital Mechanics with MATLAB

Finally, the speed at reentry V_e can be determined from

$$V_e = \sqrt{\frac{2\mu}{r_e} - \frac{\mu}{a_d}}$$

Please note that these equations are also valid for the case of de-orbit from an initial circular orbit as described in the previous `cdeorbit.m` script.

The following is a typical user interaction with this script.

```
program edeorbit
< single impulse deorbit from elliptical orbits >

please input the perigee altitude (kilometers)
? 285.798

please input the apogee altitude (kilometers)
? 35785.922

please input the entry altitude (kilometers)
? 111.252

please input the entry flight path angle (degrees)
? -4
```

The following is the script output created for this example.

```
program edeorbit
< single impulse deorbit from elliptical orbits >
initial orbit
perigee altitude      285.798000 kilometers
apogee altitude      35785.922000 kilometers
semimajor axis       24414.000000 kilometers
eccentricity         0.727044
entry altitude       111.252000 kilometers
entry fpa            -4.000000 degrees

entry trajectory
semimajor axis       24308.08290588 kilometers
```

Orbital Mechanics with MATLAB

```

eccentricity                0.73456961
perigee altitude            73.96381175 kilometers
apogee altitude            35785.92200000 kilometers
entry true anomaly         350.55084585 degrees
entry velocity             10317.40933180 meters/second
entry fpa                  -4.00000000 degrees
impulse-to-entry time      312.58844372 minutes
deorbit delta-v            22.29796787 meters/second
  
```

The software will also calculate and display the entry velocity and flight path angle relative to a rotating Earth. The following is the relative flight path information for this example.

```

relative flight path coordinates
entry velocity              9845.40345708 meters/second
entry fpa                  -4.19210209 degrees
  
```

This MATLAB script will also graphically display the initial elliptic orbit and the de-orbit trajectory.

assist.m – aero-assisted orbital transfer

This MATLAB script can be used to estimate the propulsive ΔV required for aero-assisted coplanar orbital transfer from a high Earth orbit (HEO) to a lower Earth orbit (LEO). Both the initial and final orbits are circular. The equations used in this algorithm are described in “Fuel-Optimal Trajectories for Aeroassisted Coplanar Orbital Transfer Problem”, *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 26, No. 2, March 1990, pg. 374-380.

The *normalized* delta-Vs required to initiate the aeropass $\Delta\tilde{V}_d$ and to re-circularize the orbit after the aeropass $\Delta\tilde{V}_c$ are given by

$$\Delta\tilde{V}_d = \sqrt{\frac{1}{a_d}} - \sqrt{\frac{2(1-a_d)}{a_d \left(1 - \frac{a_d^2}{\cos^2 \gamma_{entry}}\right)}}$$

$$\Delta\tilde{V}_c = \sqrt{\frac{1}{a_c}} - \sqrt{\frac{2(1-a_c)}{a_c \left(1 - \frac{a_c^2}{\cos^2 \gamma_{exit}}\right)}}$$

Orbital Mechanics with MATLAB

The normalized speeds at entry into the atmosphere \tilde{V}_{entry} and at exit from the atmosphere \tilde{V}_{exit} are given by

$$\tilde{V}_{entry} = \sqrt{\frac{2a_d(1-a_d)}{\cos^2 \gamma_{entry} - a_d^2}}$$

$$\tilde{V}_{exit} = \sqrt{\frac{2a_c(1-a_c)}{\cos^2 \gamma_{exit} - a_c^2}}$$

where

$$a_d = \frac{r_d}{r_a} = \text{initial orbit radius ratio}$$

$$a_c = \frac{r_c}{r_a} = \text{final orbit radius ratio}$$

r_d = geocentric radius of the initial orbit

r_c = geocentric radius of the final orbit

r_a = geocentric radius of the atmosphere

γ_{entry} = flight path angle at atmospheric entry

γ_{exit} = flight path angle at atmospheric exit

The dimensional speed and ΔV can be recovered by multiplying the normalized values by $\sqrt{\mu/r_a}$ where μ is the gravitational constant of the Earth.

The following is a typical user interaction with this script.

```
program assist

< aeroassisted orbit transfer between circular orbits >

please input the initial altitude (kilometers)
? 35786

please input the final altitude (kilometers)
? 300

please input the entry altitude (kilometers)
? 120

please input the entry flight path angle (degrees)
? -3

please input the exit flight path angle (degrees)
? 1
```

Orbital Mechanics with MATLAB

The following is the script output for this example.

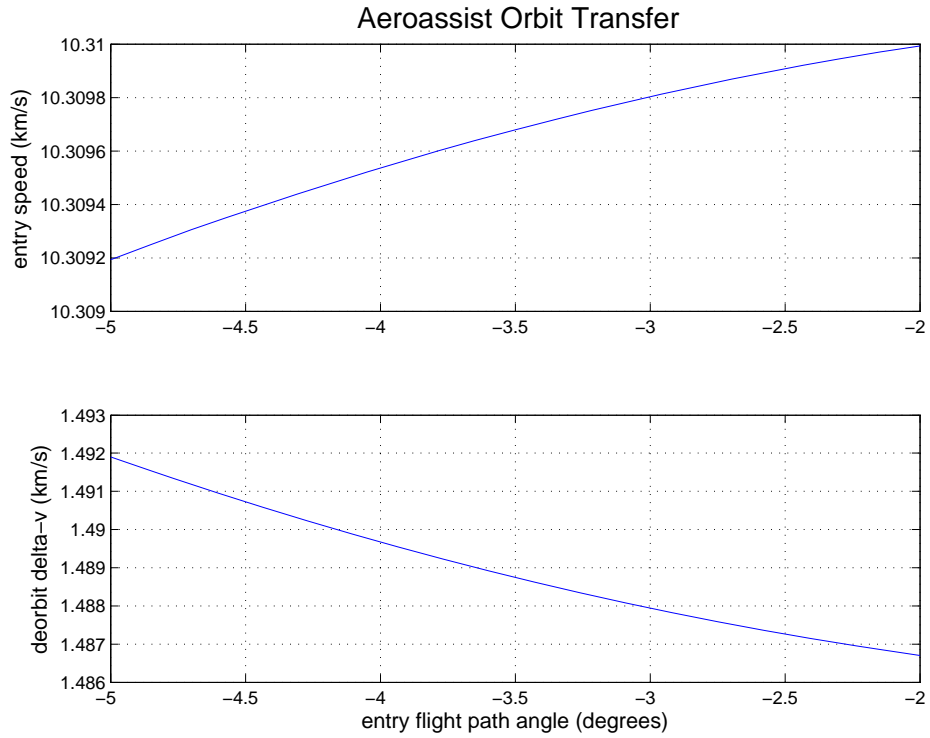
```
aeroassist orbit transfer
initial altitude          35786.0000 kilometers
final altitude           300.0000 kilometers
entry flight path angle  -3.0000 degrees
entry speed              10309.8017 meters/second
exit flight path angle    1.0000 degrees
exit speed               7864.0519 meters/second
deorbit delta-v          1487.9405 meters/second
circularization delta-v  74.8372 meters/second
total delta-v            1562.7777 meters/second
```

For comparison purposes the software will also display the all-propulsive Hohmann orbit transfer for this example.

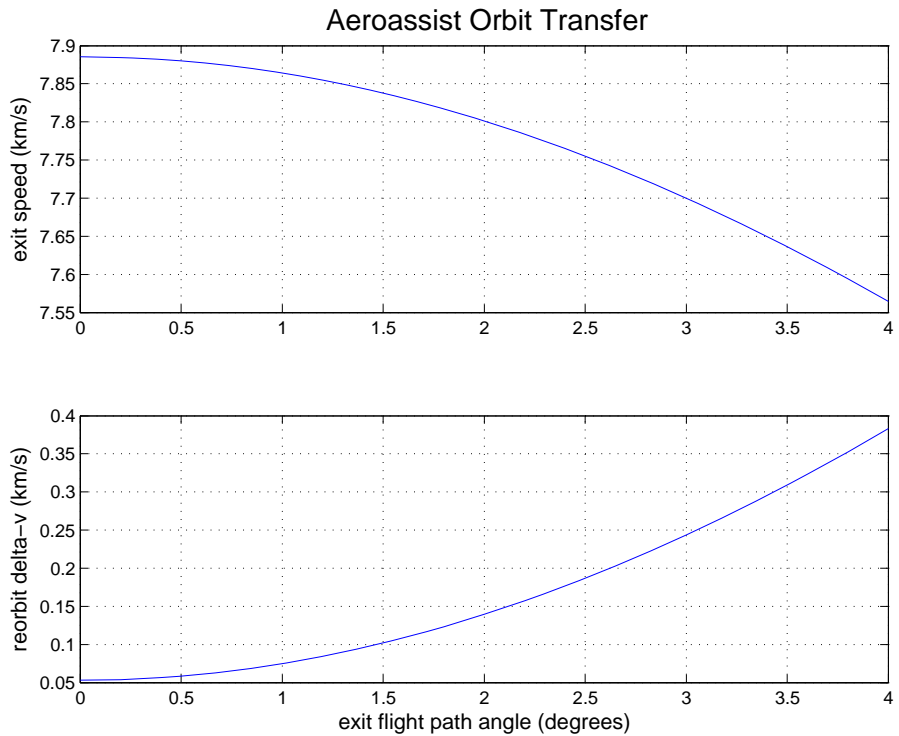
```
Hohmann orbit transfer
deorbit delta-v          1466.8241 meters/second
circularization delta-v  2425.7315 meters/second
total delta-v            3892.5557 meters/second
```

The following is a plot of the entry speed and de-orbit ΔV as a function of the flight path angle at entry into the Earth's atmosphere.

Orbital Mechanics with MATLAB



The following is a plot of the exit speed and circularization ΔV as a function of the flight path angle at exit from the atmosphere.



The Hohmann Orbit Transfer

This section summarizes the equations that define the Hohmann orbit transfer and describes a MATLAB script that solves this classic astrodynamics problem. The software can solve both the coplanar and non-coplanar orbit transfer problem.

hohmann.m – Hohmann two impulse orbital transfer

The coplanar circular orbit-to-circular orbit transfer was discovered by the German engineer Walter Hohmann in 1925 and described in his book, *The Attainability of Celestial Bodies*. The transfer consists of a velocity impulse on an initial circular orbit, in the direction of motion and collinear with the velocity vector, which propels the space vehicle into an elliptical transfer orbit. At a transfer angle of 180 degrees from the first impulse, a second velocity impulse or ΔV , also collinear and in the direction of motion, places the vehicle into a final circular orbit at the desired final altitude. The impulsive ΔV assumption means that the velocity, but not the position, of the vehicle is changed instantaneously. This is equivalent to a rocket engine with infinite thrust magnitude. Therefore, the Hohmann formulation is the ideal and minimum energy solution to this type of orbit transfer problem.

Coplanar Equations

For the coplanar Hohmann transfer both velocity impulses are confined to the orbital planes of the initial and final orbits. The first impulse creates an elliptical transfer orbit with a perigee altitude equal to the altitude of the initial circular orbit and an apogee altitude equal to the altitude of the final orbit. The second impulse circularizes the transfer orbit at apogee. Both impulses are *prograde* which means that they are in the direction of orbital motion.

We begin by defining three *normalized* radii as follows:

$$R_1 = \sqrt{2 \frac{r_f}{r_i + r_f}}$$

$$R_2 = \sqrt{\frac{r_i}{r_f}}$$

$$R_3 = \sqrt{2 \frac{r_i}{r_i + r_f}}$$

where r_i is the geocentric radius of the initial circular park orbit and r_f is the radius of the final circular mission orbit. The relationship between radius and initial orbit altitude h_i and the final orbit altitude h_f is as follows:

$$r_i = r_e + h_i$$

$$r_f = r_e + h_f$$

where r_e is the radius of the Earth.

The magnitude of the first impulse is

$$\Delta V_1 = V_{lc} \sqrt{1 + R_1^2 - 2R_1}$$

and is simply the difference between the speed on the initial orbit and the perigee speed of the transfer orbit. The scalar magnitude of the second impulse is

$$\Delta V_2 = V_{lc} \sqrt{R_2^2 + R_2^2 R_3^2 - 2R_2^2 R_3}$$

which is the difference between the speed on the final orbit and the apogee speed of the transfer ellipse. In each of these ΔV equations V_{lc} is called the *local circular velocity*. It can be determined from

$$V_{lc} = \sqrt{\frac{\mu}{r_i}}$$

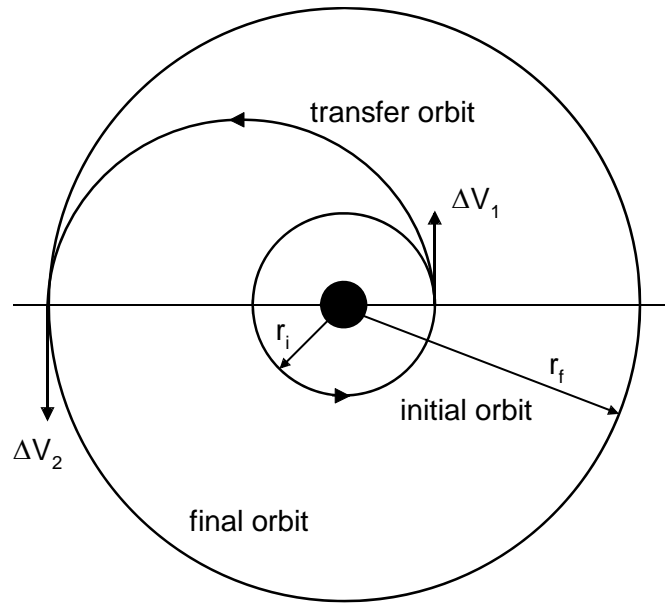
and represents the scalar speed in the initial orbit. In these equations μ is the gravitational constant of the central body. The transfer time from the first impulse to the second is equal to one half the orbital period of the transfer ellipse

$$\tau = \pi \sqrt{\frac{a^3}{\mu}}$$

where a is the semimajor axis of the transfer orbit and is equal to $(r_i + r_f)/2$. The orbital eccentricity of the transfer ellipse is

$$e = \frac{\max(r_i, r_f) - \min(r_i, r_f)}{r_f - r_i}$$

The following diagram illustrates the geometry of the coplanar Hohmann transfer.



Non-coplanar Equations

The non-coplanar Hohmann transfer involves orbital transfer between two circular orbits which have different orbital inclinations. For this problem the propulsive energy is minimized if we optimally partition the total orbital inclination change between the first and second impulses. The scalar magnitude of the first impulse is

$$\Delta V_1 = V_{lc} \sqrt{1 + R_1^2 - 2R_1 \cos \theta_1}$$

where θ_1 is the plane change associated with the first impulse. The magnitude of the second impulse is

$$\Delta V_2 = V_{lc} \sqrt{R_2^2 + R_2^2 R_3^2 - 2R_2^2 R_3 \cos \theta_2}$$

where θ_2 is the plane change associated with the second impulse. These two equations are different forms of the law of cosines.

The total ΔV required for the maneuver is the sum of the two impulses as follows

$$\Delta V = \Delta V_1 + \Delta V_2$$

The relationship between the plane change angles is

$$\theta_t = \theta_1 + \theta_2$$

where θ_t is the total plane change angle between the initial and final orbits.

Optimizing the non-coplanar Hohmann transfer involves allocating the total plane change angle between the two maneuvers such that the total ΔV required for the mission is minimized. We can determine this answer by solving for the root of a derivative.

The partial derivative of the total ΔV with respect to the first plane change angle is given by:

$$\frac{\partial \Delta V}{\partial \theta_1} = \frac{R_1 \sin \theta_1}{\sqrt{1 + R_1^2 - 2R_1 \cos \theta_1}} - \frac{R_2^2 R_3 (\sin \theta_t \cos \theta_1 - \cos \theta_t \sin \theta_1)}{\sqrt{R_2^2 + R_2^2 R_3^2 - 2R_2^2 R_3 \cos(\theta_t - \theta_1)}}$$

If we determine the value of θ_1 which makes this derivative zero, we have found the optimum plane change required at the first impulse. Once θ_1 is calculated we can determine θ_2 from the total plane change angle relationship and the velocity impulses from the previous equations.

Numerical Solution

This numerical algorithm has been implemented in an interactive MATLAB script called `hohmann.m`. This script prompts the user for the initial and final altitudes in kilometers and the initial and final orbital inclinations in degrees. The software then calls the Brent root-finding algorithm to solve the partial derivative equation described above.

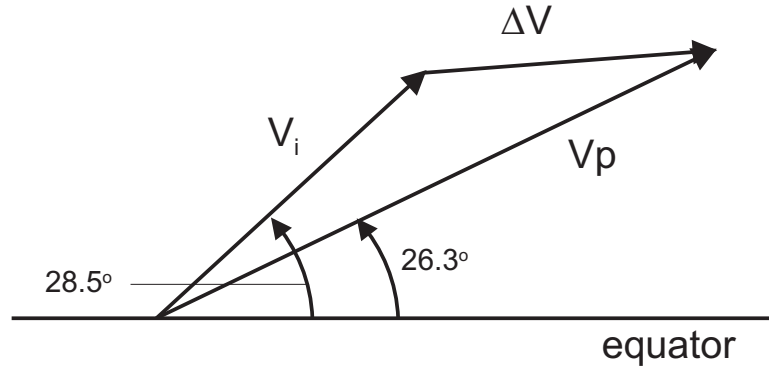
The call to the Brent root-finding algorithm is as follows:

```
[xroot, froot] = brent'hohmfunc', 0, dinc, rtol
```

where `hohmfunc` is the objective function for this problem. Since we know that the optimum first plane change angle is somewhere between 0 and the total plane change angle `dinc`, we pass these as the bounds of the root. In the parameter list `rtol` is the user-defined root-finding convergence tolerance.

This is a typical orbit transfer from a low altitude Earth orbit (LEO) at an altitude of 185.2 kilometers and an orbital inclination of 28.5 degrees to a geosynchronous Earth orbit (GEO) at an altitude of 35786.36 kilometers and 0 degrees inclination.

The following is a ΔV diagram for the first maneuver of this orbit transfer example. In this view we are looking along the line of nodes which is the mutual intersection of the park and transfer orbit planes with the equatorial plane.



In this diagram V_i is the speed on the initial park orbit, V_p is the perigee speed of the elliptic transfer orbit, and ΔV is the ΔV required for the first maneuver. The inclinations of the park and transfer orbit are also labeled. From this geometry and the law of cosines, the required ΔV is given by

$$\Delta V = \sqrt{V_i^2 + V_p^2 - 2V_i V_p \cos \Delta i}$$

where Δi is the inclination difference or plane change between the park and transfer orbits.

The following is a typical user interaction with this script.

```
Hohmann Orbit Transfer Analysis
```

```
please input the initial altitude kilometers
? 300
```

```
please input the final altitude kilometers
? 35786.2
```

```
please input the initial orbital inclination degrees
(0 <= inclination <= 180)
? 28.5
```

```
please input the final orbital inclination degrees
(0 <= inclination <= 180)
? 0
```

```
Hohmann Orbit Transfer Analysis
```

```
initial orbit altitude           300.0000 kilometers
initial orbit inclination        28.5000 degrees
initial orbit velocity           7725.7606 meters/second
```

Orbital Mechanics with MATLAB

final orbit altitude	35786.2000 kilometers
final orbit inclination	0.0000 degrees
final orbit velocity	3074.6540 meters/second
first inclination change	2.2002 degrees
second inclination change	26.2998 degrees
total inclination change	28.5000 degrees
first delta-v	2449.4551 meters/second
second delta-v	1781.8532 meters/second
total delta-v	4231.3083 meters/second
transfer orbit eccentricity	0.72654389
transfer orbit perigee velocity	10151.4962 meters/second
transfer orbit apogee velocity	1607.8298 meters/second

Primer Vector Analysis

This section describes a MATLAB script named `primer.m` that demonstrates how to use primer vector theory to analyze the performance of impulsive orbital transfers. The term primer vector was invented by Derek F. Lawden and represents the adjoint vector for velocity. A technical discussion about primer theory can be found in Lawden's classic text, *Optimal Trajectories for Space Navigation*, Butterworths, London, 1963. Another excellent resource is "Primer Vector Theory and Applications", Donald J. Jezewski, NASA TR R-454, November 1975, along with "Optimal, Multi-burn, Space Trajectories", also by Jezewski.

As shown by Lawden, the following four necessary conditions must be satisfied in order for an impulsive orbital transfer to be *locally optimal*:

- (1) the primer vector and its first derivative are everywhere continuous
- (2) whenever a velocity impulse occurs, the primer is a unit vector aligned with the impulse and has unit magnitude ($\mathbf{p} = \hat{\mathbf{p}} = \hat{\mathbf{u}}_T$ and $\|\mathbf{p}\| = 1$)
- (3) the magnitude of the primer vector may not exceed unity on a coasting arc ($\|\mathbf{p}\| = p \leq 1$)
- (4) at all interior impulses (not at the initial or final times) $\mathbf{p} \cdot \dot{\mathbf{p}} = 0$; therefore, $d\|\mathbf{p}\|/dt = 0$ at the intermediate impulses

Furthermore, the scalar magnitudes of the primer vector derivative at the initial and final impulses provide information about how to improve the nominal transfer trajectory by changing the endpoint times and/or moving the impulse times. These four cases for non-zero slopes are summarized as follows;

- If $\dot{p}_0 > 0$ and $\dot{p}_f < 0 \rightarrow$ perform an initial coast before the first impulse and add a final coast after the second impulse
- If $\dot{p}_0 > 0$ and $\dot{p}_f > 0 \rightarrow$ perform an initial coast before the first impulse and move the second impulse to a later time
- If $\dot{p}_0 < 0$ and $\dot{p}_f < 0 \rightarrow$ perform the first impulse at an earlier time and add a final coast after the second impulse
- If $\dot{p}_0 < 0$ and $\dot{p}_f > 0 \rightarrow$ perform the first impulse at an earlier time and move the second impulse to a later time

The primer vector analysis of a two impulse orbital transfer involves the following steps.

First partition the two-body state transition matrix as follows:

$$\Phi(t, t_0) = \begin{bmatrix} \frac{\partial \mathbf{r}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{r}}{\partial \mathbf{v}_0} \\ \frac{\partial \mathbf{v}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{v}}{\partial \mathbf{v}_0} \end{bmatrix} = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} = \begin{bmatrix} \Phi_{rr} & \Phi_{rv} \\ \Phi_{vr} & \Phi_{vv} \end{bmatrix}$$

where

$$\Phi_{11} = \begin{bmatrix} \frac{\partial \mathbf{r}}{\partial \mathbf{r}_0} \end{bmatrix} = \begin{bmatrix} \partial x / \partial x_0 & \partial x / \partial y_0 & \partial x / \partial z_0 \\ \partial y / \partial x_0 & \partial y / \partial y_0 & \partial y / \partial z_0 \\ \partial z / \partial x_0 & \partial z / \partial y_0 & \partial z / \partial z_0 \end{bmatrix}$$

and so forth.

The value of the primer vector at any time t along a two body trajectory is given by

$$\mathbf{p}(t) = \Phi_{11}(t, t_0)\mathbf{p}_0 + \Phi_{12}(t, t_0)\dot{\mathbf{p}}_0$$

and the value of the primer vector derivative is

$$\dot{\mathbf{p}}(t) = \Phi_{21}(t, t_0)\mathbf{p}_0 + \Phi_{22}(t, t_0)\dot{\mathbf{p}}_0$$

which can also be expressed as

$$\begin{Bmatrix} \mathbf{p} \\ \dot{\mathbf{p}} \end{Bmatrix} = \Phi(t, t_0) \begin{Bmatrix} \mathbf{p}_0 \\ \dot{\mathbf{p}}_0 \end{Bmatrix}$$

The primer vector boundary conditions at the initial and final impulses are as follows:

$$\mathbf{p}(t_0) = \mathbf{p}_0 = \frac{\Delta \mathbf{V}_0}{|\Delta \mathbf{V}_0|} \quad \mathbf{p}(t_f) = \mathbf{p}_f = \frac{\Delta \mathbf{V}_f}{|\Delta \mathbf{V}_f|}$$

These two conditions illustrate that at the locations of velocity impulses, the primer vector is a unit vector in the direction of the impulses.

The value of the primer vector derivative at the initial time is

$$\dot{\mathbf{p}}(t_0) = \dot{\mathbf{p}}_0 = \Phi_{12}^{-1}(t_f, t_0) \{ \mathbf{p}_f - \Phi_{11}(t_f, t_0) \mathbf{p}_0 \}$$

provided the Φ_{12} sub-matrix is non-singular. The value of the primer vector derivative at the final time is

$$\dot{\mathbf{p}}(t_f) = \dot{\mathbf{p}}_f = \Phi_{21}(t_f, t_0) \mathbf{p}_0 + \Phi_{22}(t_f, t_0) \Phi_{12}^{-1}(t_f, t_0) \{ \mathbf{p}_f - \Phi_{11}(t_f, t_0) \mathbf{p}_0 \}$$

The scalar magnitude of the derivative of the primer vector can be determined from

$$\frac{d\|\mathbf{p}\|}{dt} = \frac{d}{dt}(\mathbf{p} \cdot \mathbf{p})^2 = \frac{\dot{\mathbf{p}} \cdot \mathbf{p}}{\|\mathbf{p}\|}$$

This MATLAB script creates plots of the scalar magnitudes of the primer vector and its derivative for a two impulse, coplanar Hohmann transfer. It will request the altitudes of the initial and final circular orbits.

The following is a typical user interaction with this script.

```
Primer Vector Analysis

please input the initial altitude (kilometers)
? 185.2

please input the final altitude (kilometers)
? 35786
```

An example of the output produced by this application is

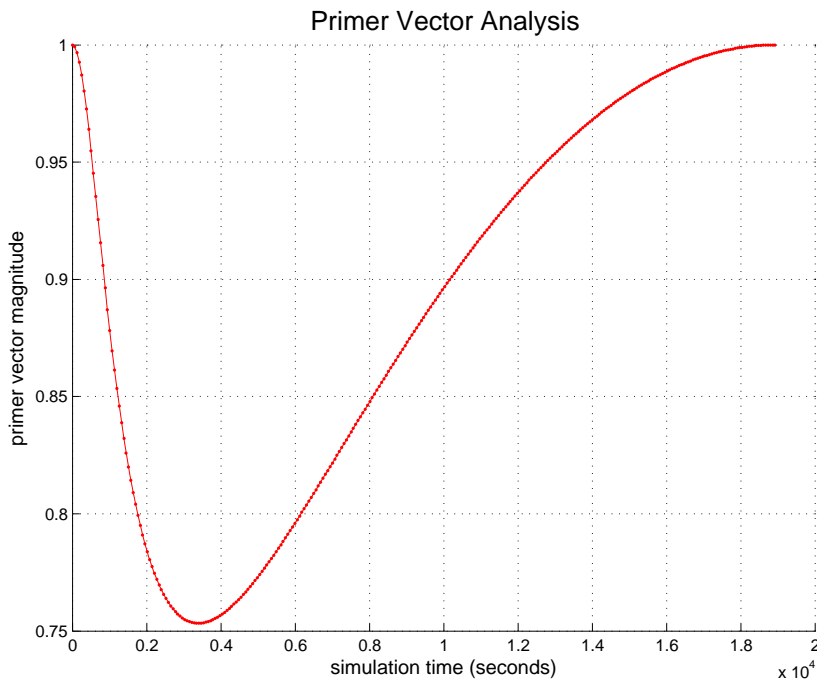
```
Primer Vector Analysis

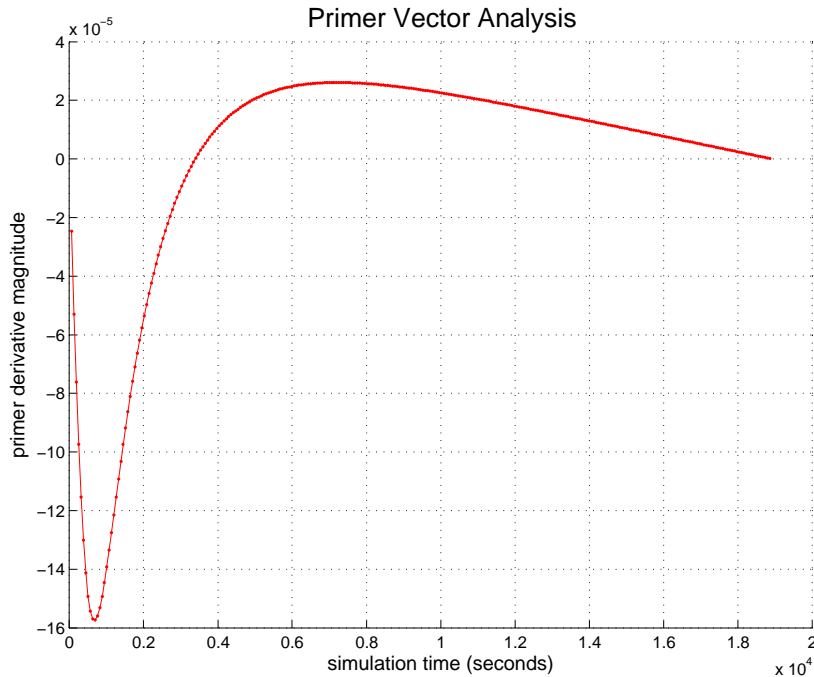
initial orbit altitude           185.2000 kilometers
```

Orbital Mechanics with MATLAB

initial orbit inclination	0.0000 degrees
initial orbit velocity	7793.0337 meters/second
final orbit altitude	35786.0000 kilometers
final orbit inclination	0.0000 degrees
final orbit velocity	3074.6613 meters/second
first delta-v	2458.9100 meters/second
second delta-v	1478.8275 meters/second
total delta-v	3937.7375 meters/second
transfer orbit eccentricity	0.73061044
transfer orbit perigee velocity	10251.9438 meters/second
transfer orbit apogee velocity	1595.8338 meters/second
transfer time-of-flight	18923.2978 seconds

The following are plots of the scalar magnitude of the primer vector and primer derivative as a function of elapsed trajectory time for this coplanar orbital transfer. From these plots and the necessary conditions for optimality, we can see that this is a locally optimal two impulse transfer trajectory since $p_0 = p_f = 1$.





From this plot of the primer derivative magnitude ($\dot{p}_0 = \dot{p}_f \approx 0$) and the rules for moving impulses or adding coasts, no modifications of this trajectory will reduce the total delta-v.

Phasing Analysis

This section describes a MATLAB script called `phasing.m` that performs a phasing or rendezvous analysis accomplished with a two impulse, coplanar Hohmann transfer. This script computes and displays a complete analysis of the phasing maneuvers along with a graphics display of the initial orbit and final orbits, the transfer trajectory and the orbital locations of the spacecraft and maneuvers. In this script, the spacecraft on the inner circular orbit is called the *chaser* and the spacecraft on the outer circular orbit is called the *target*.

The `phasing.m` script will prompt the user for the altitudes of the initial and final circular orbits, the initial true anomaly of the chaser vehicle and the initial lead angle of the target vehicle relative to the chaser. This initial lead angle can be either the “ideal” Hohmann value computed by the script or a user-defined value.

The following a typical interaction with this script.

```
TWO IMPULSE PHASING ANALYSIS

please input the initial altitude (kilometers)
? 500

please input the true anomaly on the initial orbit (degrees)
(0 <= true anomaly <= 360)
```

Orbital Mechanics with MATLAB

? 45

please input the final altitude (kilometers)

? 5000

type of initial lead angle

<1> ideal

<2> user-defined

selection (1 or 2)

? 2

please input the initial lead angle (degrees)

? 45

The following is the script output for this example.

```
TWO IMPULSE PHASING ANALYSIS

initial lead angle          45.000000 degrees
ideal initial lead angle   50.658176 degrees

chaser initial conditions
-----
altitude          500.000000 kilometers

      sma (km)          eccentricity          inclination (deg)          argper (deg)
+6.87813630000000e+003  +0.00000000000000e+000  +0.00000000000000e+000  +0.00000000000000e+000

      raan (deg)          true anomaly (deg)          arglat (deg)          period (min)
+0.00000000000000e+000  +4.50000000000000e+001  +4.50000000000000e+001  +9.46162866922553e+001

      rx (km)          ry (km)          rz (km)          rmag (km)
+4.86357681965535e+003  +4.86357681965535e+003  +0.00000000000000e+000  +6.87813630000000e+003

      vx (kps)          vy (kps)          vz (kps)          vmag (kps)
-5.38292709812771e+000  +5.38292709812772e+000  +0.00000000000000e+000  +7.61260850743786e+000

target initial conditions
-----
altitude          5000.000000 kilometers

      sma (km)          eccentricity          inclination (deg)          argper (deg)
+1.13781363000000e+004  +0.00000000000000e+000  +0.00000000000000e+000  +0.00000000000000e+000

      raan (deg)          true anomaly (deg)          arglat (deg)          period (min)
+0.00000000000000e+000  +9.00000000000000e+001  +9.00000000000000e+001  +2.01310501632031e+002

      rx (km)          ry (km)          rz (km)          rmag (km)
+6.96709910002865e-013  +1.13781363000000e+004  +0.00000000000000e+000  +1.13781363000000e+004

      vx (kps)          vy (kps)          vz (kps)          vmag (kps)
-5.91879528089421e+000  +3.62421684777778e-016  +0.00000000000000e+000  +5.91879528089421e+000
```

Orbital Mechanics with MATLAB

transfer orbit initial conditions

sma (km)	eccentricity	inclination (deg)	argper (deg)
+9.12813630000000e+003	+2.46490622625782e-001	+0.00000000000000e+000	+3.53571678494045e+002
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
+0.00000000000000e+000	+0.00000000000000e+000	+3.53571678494045e+002	+1.44654819447959e+002
rx (km)	ry (km)	rz (km)	rmag (km)
+6.83489138174017e+003	-7.70077113795533e+002	+0.00000000000000e+000	+6.87813630000000e+003
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
+9.51571536023532e-001	+8.44576208559208e+000	+0.00000000000000e+000	+8.49919911489282e+000

trajectory times

transfer time-of-flight	4339.644583 seconds 72.327410 minutes
wait time	10542.961601 seconds 175.716027 minutes
total mission time	14882.606185 seconds 248.043436 minutes

conditions at first impulse

chaser true anomaly	353.571678 degrees
target true anomaly	44.229854 degrees
transfer orbit true anomaly	0.000000 degrees

conditions at second impulse

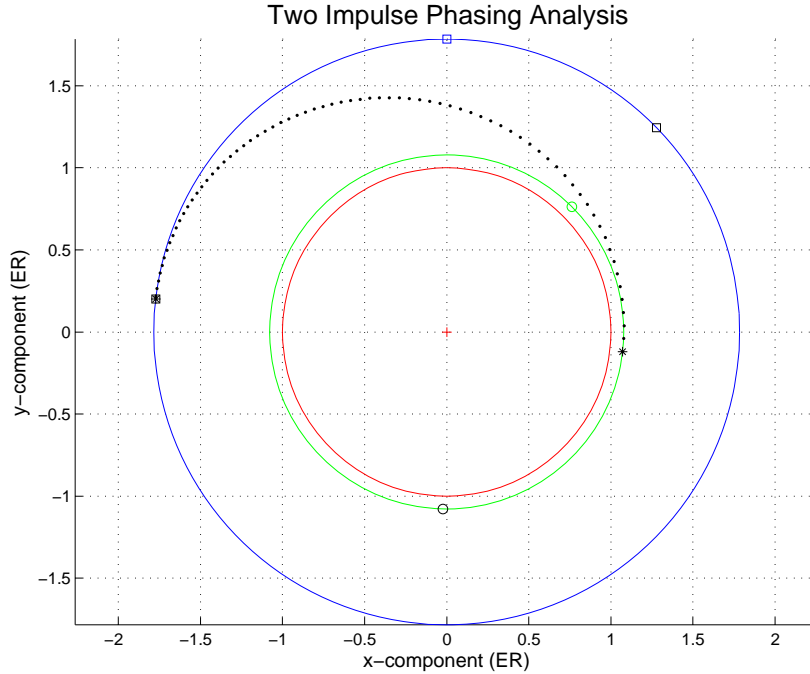
chaser true anomaly	268.766028 degrees
target true anomaly	173.571678 degrees
transfer orbit true anomaly	180.000000 degrees

maneuver summary

first delta-vx	99.262810 meters/second
first delta-vy	881.016345 meters/second
first delta-vz	0.000000 meters/second
first deltav-magnitude	886.590607 meters/second
second delta-vx	-87.439739 meters/second
second delta-vy	-776.079577 meters/second
second delta-vz	0.000000 meters/second
second deltav-magnitude	780.989896 meters/second
total delta-v	1667.580503 meters/second

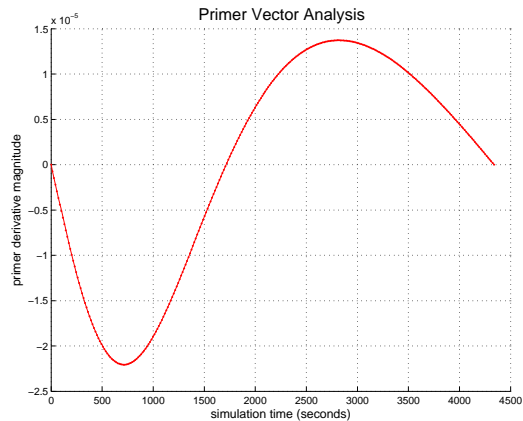
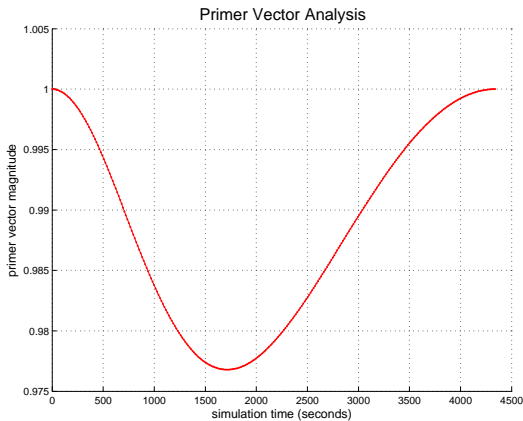
Orbital Mechanics with MATLAB

The following is the graphics output for this example. In this plot, the red circle is the surface of the spherical Earth, the green circle is the initial orbit and the blue circle is the final orbit. The black trace is the transfer trajectory at 60 second intervals. The small green circle is the initial location of the chaser vehicle and the small blue square is the initial location of the target vehicle. The small black square is the location of the target vehicle at the time of the first impulse and the small black circle is the location of the chaser vehicle at the time of the second impulse. The two black asterisks are the locations of the two velocity impulses. They are also the locations of the chaser and target vehicles at the maneuver times.



This script also performs a graphical primer vector analysis of the transfer trajectory. Please see the discussion in the previous Primer Vector Analysis section for information about this feature.

Here are the plots of the primer vector and its derivative for this example.



Orbital Mechanics with MATLAB

The “ideal” Hohmann lead angle is the lead angle the target vehicle must have relative to the chaser vehicle in order for the 180° transfer to intercept the target. It can be computed from

$$\beta_H = \pi \left\{ 1 - \left[(1 + R) / 2R \right]^{3/2} \right\}$$

where

$$R = r_1 / r_2$$

r_1 = radius of the chaser circular orbit

r_2 = radius of the target circular orbit

If the initial lead angle is not equal to the “ideal” Hohmann value, a waiting time is required before the first impulsive maneuver can be performed. This wait time is given by

$$t_{wait} = \frac{\Delta\beta\tau_s}{2\pi}$$

where

$$\Delta\beta = \beta - \beta_H$$

β = user-defined initial lead angle

τ_s = synodic period of chaser/target orbits

The synodic period is computed from the orbital periods of the chaser and target orbits using the following equation;

$$\tau_s = \frac{\tau_1\tau_2}{\tau_2 - \tau_1}$$

In this expression, τ_1 is the orbital period of the chaser vehicle and τ_2 is the orbital period of the target vehicle. The orbital period can be computed from the circular orbit radius r according to $\tau = 2\pi\sqrt{r^3/\mu}$ where μ is the gravitational constant of the central body.

The equations for computing the impulsive delta-v vectors and magnitudes can be found in the Hohmann transfer section given earlier in this document.

Two good resources for this material are “Optimal Multiple-Impulse Time-Fixed Rendezvous Between Circular Orbits”, John E. Prussing and Jeng-Hua Chiu, *AIAA Journal of Guidance and Control*, Vol. 9, No. 1, January-February 1986, pp. 17-22, and “Transfer Between Vehicles in Circular Orbits”, Bernard H. Paiewonsky, *Jet Propulsion*, February 1958, pp.121-123.