

# Program hyper\_ocs

## Optimal Finite-Burn Earth Orbit-to-Interplanetary Injection

This document is the user's manual for a Fortran computer program called `hyper_ocs` that uses optimal control software distributed by Applied Mathematical Analysis to solve the single-maneuver, finite-burn Earth orbit-to-interplanetary injection trajectory optimization problem. The software attempts to maximize the final spacecraft mass. Since this simulation involves a single propulsive maneuver, this is equivalent to minimizing the propellant mass required for the orbital maneuver.

The important features of this scientific simulation are as follows:

- single, continuous thrust propulsive maneuver
- inertially fixed or variable attitude steering
- modified equinoctial equations of motion
- valid for circular and elliptical park orbits
- user-specified departure hyperbola targets (C3, RLA and DLA)
- user-specified final coast duration

The `AMA_OC` software suite is a *direct transcription method* that can be used to solve a variety of trajectory optimization problems using the following combination of numerical methods:

- collocation and *implicit* integration
- adaptive mesh refinement
- sparse nonlinear programming

Additional information about the mathematical techniques and numerical methods used in the `AMA_OC` software can be found in the book, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming* by John. T. Betts, SIAM, 2010 ([www.siam.org](http://www.siam.org)).

The `hyper_ocs` software consists of Fortran routines that perform the following tasks:

- set algorithm control parameters and call the transcription/optimal control subroutine
- define the problem structure and perform initialization related to scaling, lower and upper bounds, initial conditions, etc.
- compute the *right-hand-side* differential equations
- evaluate any point and path constraints
- display the optimal solution results and create an output file

The `AMA_OC` software will use this information to *automatically* transcribe the user's optimal control problem and perform the optimization using a sparse nonlinear programming (NLP) method. The `hyper_ocs` software allows the user to select the type of initial guess, collocation method, and other important algorithm control parameters.

## Program Execution

An input file created by the user can be run from the command line or a simple batch file with a statement similar to the following:

```
hyper_ocs hyper1.in
```

If the software is executed without an input file on the command line, the computer program will display the following title screen and file name prompt:

```
*****
*           Program hyper_ocs           *
*                                           *
*   Earth orbit-to-interplanetary      *
*   trajectory optimization             *
*                                           *
*           November 8, 2011           *
*****

please input the name of the simulation definition file
```

The user should respond to this prompt with the name of a compatible input data file including the filename extension.

## Input File Format and Contents

The `hyper_ocs` software is “data-driven” by a user-created text file. The following is a typical input file used by this computer program. In the following discussion the actual input file contents are in `courier` font and all explanations are in *times italic* font. Each data item within an input file is preceded by one or more lines of *annotation* text. Do not delete any of these annotation lines or change the number of lines. However, you may change them to reflect your own explanation. The annotation line also includes the correct units and when appropriate, the valid range of the input. All program inputs and outputs are in an Earth-centered-inertial (ECI) coordinate system. Park orbit angular elements and the hyperbolic targets should be specified in the same coordinate system.

*The first six lines of any input file are reserved for user comments. These lines are ignored by the software. However the input file must begin with six and only six initial text lines.*

```
*****
** Optimal Finite-Burn Earth Orbit-to-Interplanetary Injection
** single phase, continuous-thrust maneuver with user-defined final coast
** variable and fixed inertial attitude steering
** input file ==> hyper1.in - February 14, 2011
*****
```

*The first input defines the type of steering to use during the solution process. Additional information about these steering options can be found in the Problem Setup section.*

```
type of steering
1 = fixed inertial attitude
2 = variable attitude
-----
2
```

*The next two inputs allow the user to provide an initial guess for the right ascension and declination for the inertially fixed steering option.*

```
initial guess for right ascension (degrees)
0.0d0

initial guess for declination (degrees)
0.0d0
```

*The next three inputs define the initial mass, thrust magnitude and specific impulse of the propulsion system, respectively.*

```
initial spacecraft mass (kilograms)
4000.0

thrust magnitude (newtons)
19840.0

specific impulse (seconds)
450.0
```

*The next input defines the user's initial guess for the maneuver duration, in seconds.*

```
initial guess for thrust duration (seconds)
550.0
```

*The next two inputs define the lower and upper bounds for the thrust duration, also in seconds.*

```
lower bound for thrust duration (seconds)
1.0

upper bound for thrust duration (seconds)
1000.0
```

*The next six inputs define the classical orbital elements of the initial park orbit.*

```
*****
* INITIAL ORBIT *
*****

semimajor axis (kilometers)
6563.34

orbital eccentricity (non-dimensional)
0.015

orbital inclination (degrees)
28.5

argument of perigee (degrees)
90.0

right ascension of the ascending node (degrees)
0.0

true anomaly (degrees)
145.0
```

*This next integer input allows the user to define the type of initial park orbit constraints to use during the simulation. Option 1 will constrain the park orbit semimajor axis, eccentricity and orbital*

*inclination to the values input by the user. All other elements are unconstrained. Option 2 will constrain all initial elements, and option 3 will constrain all initial elements except true longitude.*

```
*****
initial orbit constraint options
*****
1 = constrain semimajor axis, eccentricity and inclination
2 = constrain all initial orbital elements
3 = option 2 with unconstrained true longitude
-----
1
```

*The next three inputs define the characteristics of the departure hyperbola. These inputs are the specific orbital energy (C3) in kilometers per second squared, and the right ascension (RLA) and declination (DLA) of the outgoing asymptote, respectively.*

```
*****
* LAUNCH HYPERBOLA *
*****

specific orbital energy (C3; [km/sec]**2)
8.788564

right ascension of outgoing asymptote (RLA; degrees)
349.68004

declination of outgoing asymptote (DLA; degrees)
-6.666253
```

*This next numeric input allows the user to specify a final fixed-duration orbit coast. For a simulation without a final coast, set this value to zero.*

```
final coast duration (seconds)
100.0
```

*This integer input specifies the type of gravity model to use during the simulation. Option 2 will include the oblateness gravity coefficient ( $J_2$ ) in the equations of motion.*

```
*****
* type of gravity model *
-----
1 = spherical Earth
2 = oblate Earth
-----
2
```

*This next input defines the type of initial guess to use. Please see the technical discussion section for information about how the first option is modeled. Option 2 requires either a binary restart file created from a previous run using either initial guess option 1 or an updated binary restart file. This feature is described in the next two sections.*

```
*****
* initial guess options *
*****
1 = numerical integration
2 = binary data file
-----
1
```

If the user elects to use a binary data file (option 3 above) for the initial guess, the following text input specifies the name of the file to use.

```
name of binary initial guess data file
hyper1.rsbin
```

The following input can be used to create or update an initial guess binary file. The creation or update process uses the filename defined above. For initial guess option 1, the software will create a binary restart file. For initial guess option 2, an input of yes to this item will update the binary file used to initialize the simulation.

```
*****
* binary restart file option *
*****

create/update binary data file (yes or no)
no
```

This next input specifies the type of comma-delimited or comma-separated-variable (CSV) solution data file to create. Option 1 will create a solution file at each collocation point or node determined by OCS. Options 2 and 3 allow the user to specify either the number of nodes or time step size used to create the data file.

```
*****
* type of comma-delimited solution data file *
*****
1 = OCS-defined nodes
2 = user-defined nodes
3 = user-defined step size
-----
1
```

For options 2 or 3, this next input defines either the number of data points or the time step size of the data written to the solution file.

```
number of user-defined nodes or print step size in solution data file
25
```

The name of the solution data file is defined in this next line. Please consult Appendix A for a description of the information written to this file.

```
name of solution output file
hyper1.csv
```

The next series of program inputs are algorithm control options and parameters for the AMA\_OC software. The first input is an integer that specifies the type of collocation method to use during the solution process. Additional information about these options can be found in the OC user's manual.

```
*****
* algorithm control parameters *
*****

discretization/collocation method
-----
1 = trapezoidal
2 = separated Hermite-Simpson
3 = compressed Hermite-Simpson
-----
1
```

*The next input is an integer that defines the number of grid points to use for the initial guess. The AMA\_OC software will use mesh refinement to change the number and distribution of the grid points.*

```
number of grid points  
25
```

*The next input defines the relative error in the objective function (performance index).*

```
relative error in the objective function (performance index)  
1.0d-5
```

*The next input defines the relative error in the solution of the differential equations.*

```
relative error in the solution of the differential equations  
1.0d-7
```

*The next input is an integer that defines the maximum number of mesh refinement iterations.*

```
maximum number of mesh refinement iterations  
20
```

*The next input is an integer that defines the maximum number of function evaluations.*

```
maximum number of function evaluations  
50000
```

*The next input is an integer that defines the maximum number of algorithm iterations.*

```
maximum number of algorithm iterations  
10000
```

*The level of output from the OC NLP algorithm is controlled with the following integer input. Additional information about these algorithm items can be found in the OC user's manual.*

```
*****  
sparse NLP iteration output  
-----  
1 = none  
2 = terse  
3 = standard  
4 = interpretive  
5 = diagnostic  
-----  
2
```

*The level of output from the AMA\_OC optimal control algorithm is controlled with the following integer input.*

```
*****  
optimal control output  
-----  
1 = none  
2 = terse  
3 = standard  
4 = interpretive  
-----  
1
```

*The level of output from the AMA\_OC differential equations algorithm is controlled with the following integer input.*

```

*****
differential equation output
-----
1 = none
2 = terse
3 = standard
4 = interpretive
5 = diagnostic
-----
1

```

The level of output can be further controlled by the user with this final text input. This program option sets the value of the SOCOUT character variable described in the AMA\_OC user's manual. To ignore this special output control, input the simple character string no.

```

*****
user-defined output
-----
input no to ignore
-----
a0b0c0d0e0f0g0h0i0j2k0l0m0n0o0p0q0r0

```

The last series of inputs allow the reading and writing of configuration input files. The user should create a configuration file before attempting to read one. These configuration files are simple text files which can be edited external to the hyper\_ocs software. Please consult Appendix C.

```

*****
* optimal control configuration options
*****

read an optimal control configuration file (yes or no)
no

name of optimal control configuration file
hyper1_config.txt

create an optimal control configuration file (yes or no)
no

name of optimal control configuration file
hyper1_config1.txt

```

## Optimal Control Solution

The following is the solution for this example. This simulation used an oblate Earth gravity model, a numerical integration initial guess, variable attitude steering, and a final 100 second coast. The program output includes the orbital characteristics at the beginning and end of the propulsive maneuver.

Please see Appendix A for additional information about the contents of this data display.

```

program hyper_ocs
=====

input file ==> hyper1.in

numerical integration initial guess

variable attitude steering

oblate earth gravity model

```

```
-----
beginning of finite burn
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.656334000000D+04	0.150000000000D-01	0.285000000000D+02	0.195006107461D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.213742358980D+01	0.340195936757D+03	0.175202044219D+03	0.881956335064D+02
rx (km)	ry (km)	rz (km)	rmag (km)
-.646112461247D+04	0.234812204181D+03	0.258243682196D+03	0.647054540425D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.362932694068D+00	-.694302305251D+01	-.375978161985D+01	0.790400196591D+01

*We can see how the software has modified the angular elements of the park orbit.*

```
-----
end of finite burn
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.453027509263D+05	0.114402245190D+01	0.284832207197D+02	0.194946919587D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.212549933630D+01	0.211148057289D+02	0.216061725316D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.533674174038D+04	-.370176984989D+04	-.189971873374D+04	0.676704098498D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.504708187457D+01	-.879765348372D+01	-.487168058542D+01	0.112518893522D+02

*The following program output is the final spacecraft mass, the propellant mass consumed, the actual thrust duration for the maneuver, and the accumulated delta-v. The delta-v is computed using a cubic spline numerical method which uses all data points of the output file.*

final mass	1764.41952920903	kilograms
propellant mass	2235.58047079097	kilograms
thrust duration	497.258056991095	seconds
	8.28763428318491	minutes
delta-v	3611.91380012394	meters/second

*This section of the output file summarizes the classical orbital elements and state vector at the end of the user-defined coast following the maneuver termination.*

```
-----
end of final coast
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.453544448786D+05	0.114386211719D+01	0.284806735260D+02	0.194956415150D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.212090855648D+01	0.301198658443D+02	0.225076280994D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.479993020204D+04	-.455676834369D+04	-.237406820531D+04	0.703133469023D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.566636372018D+01	-.829462754194D+01	-.461067843664D+01	0.110529127163D+02

*This final section of the program output displays the actual hyperbolic conditions computed by the software. From this display we can determine how well the software satisfies the user-defined targets.*

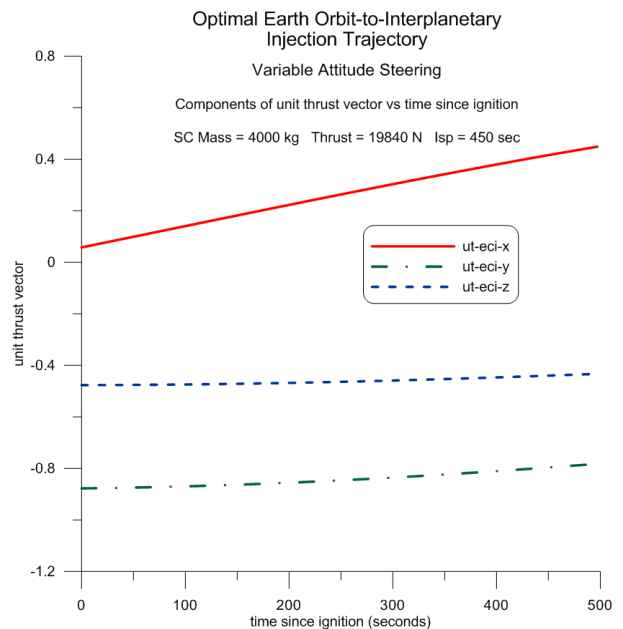
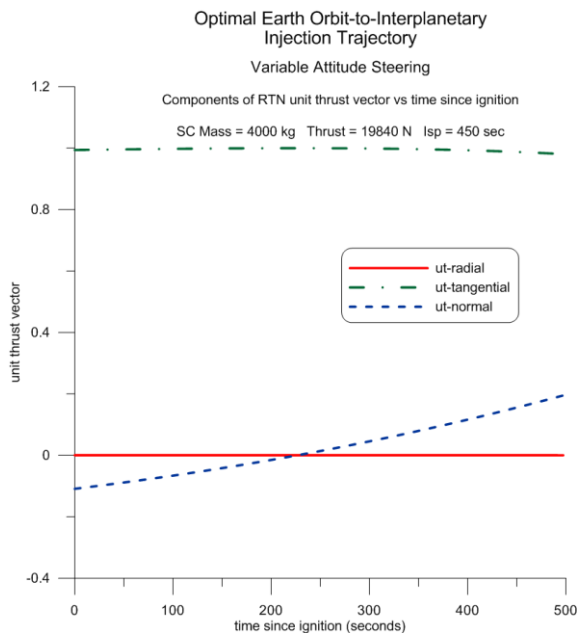
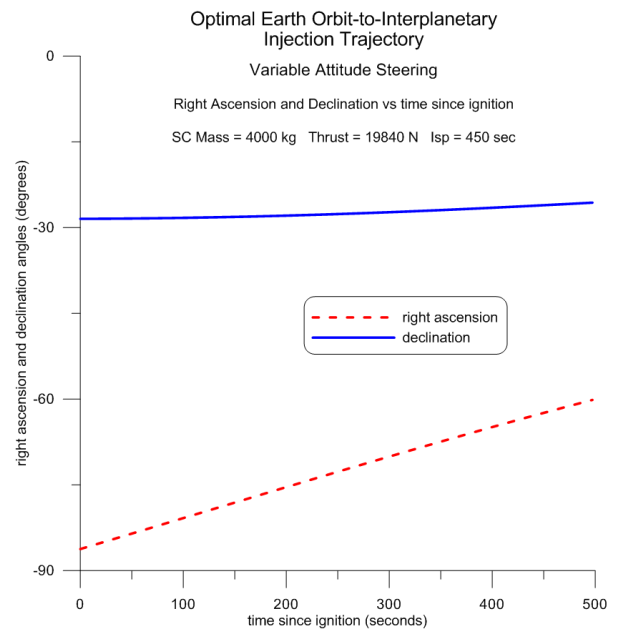
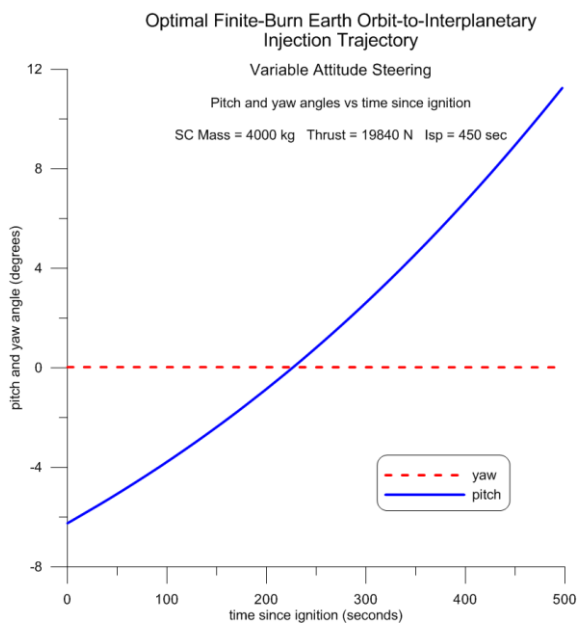
```

-----
outgoing hyperbola
-----

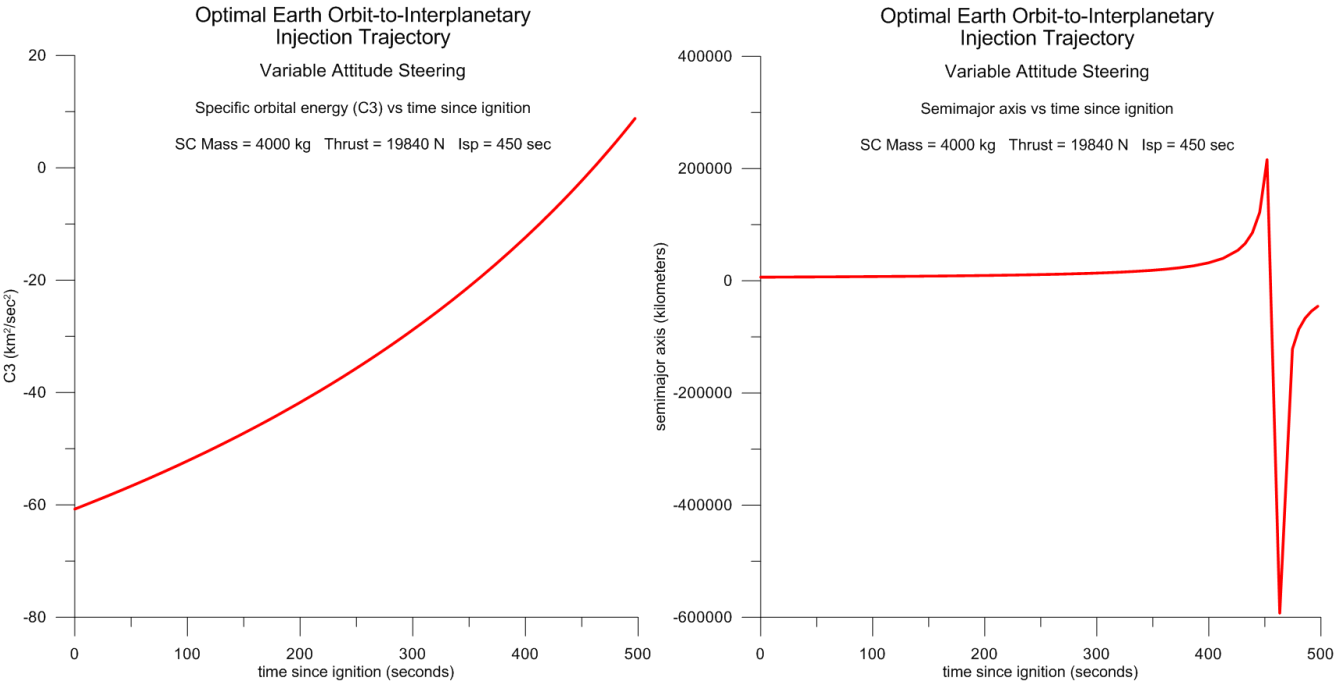
right ascension      349.680040000001      degrees
declination          -6.66625299999994      degrees
orbital energy       8.788563999999989      (km/sec)**2
final coast duration 100.000000000000      seconds
                    1.66666666666667      minutes

```

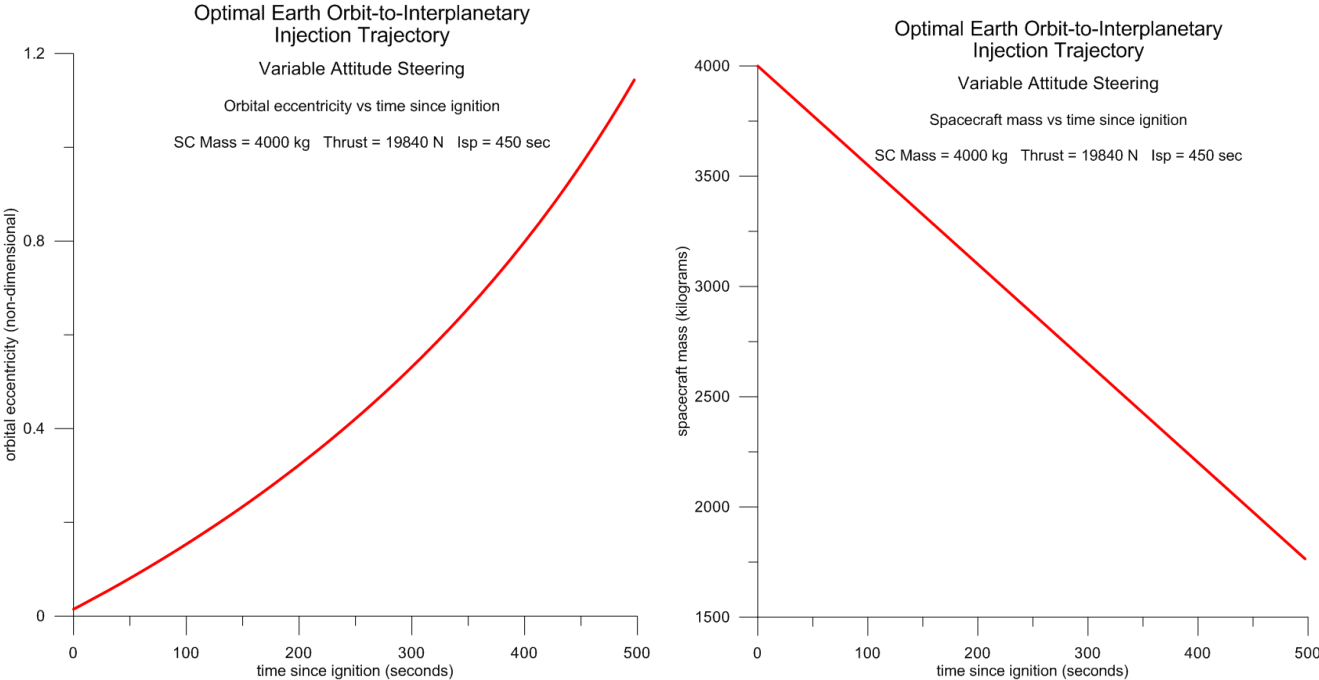
The following are plots of the behavior of the pitch and yaw angles, the inertial right ascension and declination angles, and the components of the ECI and RTN unit thrust vector during the maneuver.



The following plots illustrate the behavior of the specific orbital energy (C3) and the semimajor axis of the spacecraft's orbit as a function of time since ignition. We can see the transition from an elliptical to a hyperbolic orbit, especially the plot of semimajor axis, as orbital energy is added to the trajectory.



These final two plots illustrate the time evolution of the eccentricity of the transfer orbit and the spacecraft's instantaneous mass.

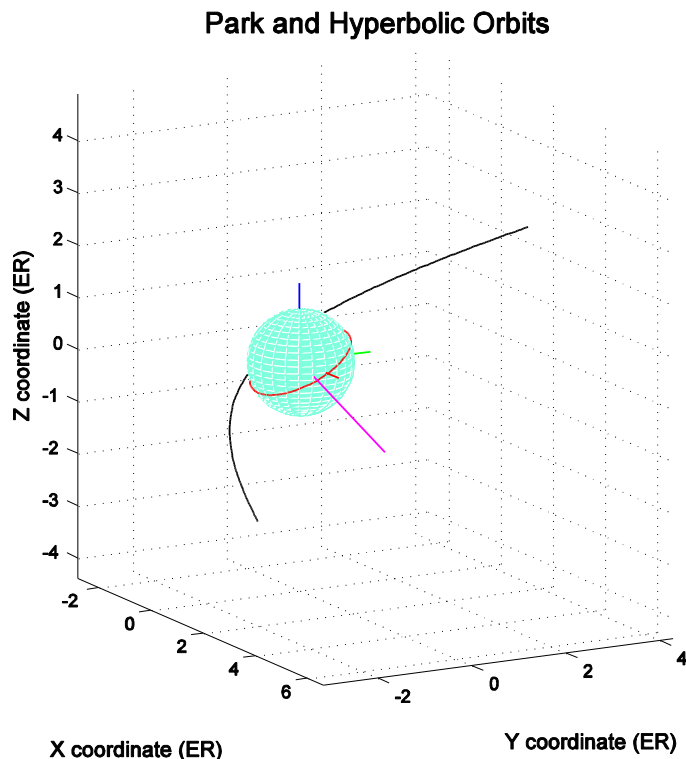


The `hyper_ocs` computer program will also create an output file called `hyper_ocs.dat`. This file contains the Earth-centered inertial position and velocity vectors of the park orbit and launch hyperbola at interplanetary injection, and the orientation of the departure asymptote. It is written as a single column of data.

The `hyper_ocs` software package includes a MATLAB script called `hplot.m` that can be used to create trajectory graphic displays using this data file. This script will interactively prompt the user for the name of the data file to use. The interactive graphic features of MATLAB allow the user to rotate and zoom the display. These capabilities allow the user to interactively find the best viewpoint as well as verify basic orbital geometry of the hyperbolic departure. The `hplot` MATLAB script will also create a color encapsulated postscript (`eps`) file (with TIFF preview) of the screen image and save it to disk with the name `hplot1.eps`. This file name can be changed by editing the final line of the `hplot` script. The user can also change the perspective by editing the `view` command on line 234 of the script. The `hplot` script and support functions were created with version 7.8 of MATLAB.

The next plot is a typical display of the park and hyperbolic orbits for this example. This display is labeled with an Earth centered, inertial coordinate system. The x-axis of this system is red, the y-axis green and the z-axis blue. The outgoing asymptote is colored magenta, the park orbit trace is red, and the hyperbolic trajectory is black.

The fundamental plane of this right-handed, orthogonal coordinate system is the Earth's equator. The x-axis points in the direction of the Vernal Equinox, the z-axis is aligned with the Earth's spin axis, and the y-axis is advanced 90 degrees along the equator from the x-axis. As mentioned earlier, the classical orbital elements of the park orbit and the hyperbolic targets (C3, RLA and DLA) should be provided in the same ECI coordinate system, perhaps true-of-date or Earth mean equator and equinox of J2000.



## Fixed inertial attitude solution

This section summarizes the output for this same example with fixed inertial attitude steering. The output includes the optimized right ascension and declination angles. The initial guess for the right ascension and declination angles were both set to zero.

-----  
beginning of finite burn  
-----

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.656334000000D+04	0.150000000000D-01	0.285000000000D+02	0.194942230990D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.213772655895D+01	0.339972859914D+03	0.174915090905D+03	0.881956335064D+02
rx (km)	ry (km)	rz (km)	rmag (km)
-.645952149379D+04	0.263240915368D+03	0.273654514483D+03	0.647067233767D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.401992811987D+00	-.694155708635D+01	-.375819334287D+01	0.790384907537D+01

-----  
end of finite burn  
-----

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.453006589465D+05	0.114267116923D+01	0.284827756905D+02	0.194823732303D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.212595426613D+01	0.214243278329D+02	0.216248060135D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.527858325139D+04	-.368582862394D+04	-.189218731497D+04	0.671037604176D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.507174069203D+01	-.883000610414D+01	-.488964831758D+01	0.112960268288D+02
final mass	1754.15735504550	kilograms	
propellant mass	2245.84264495450	kilograms	
thrust duration	499.540662719961	seconds	
	8.32567771199936	minutes	
delta-v	3637.65698948535	meters/second	

inertial steering angles

right ascension	-71.2198741186687	degrees
declination	-27.4489611164256	degrees

-----  
end of final coast  
-----

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.453544448786D+05	0.114250588430D+01	0.284801507703D+02	0.194833467431D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.212118370349D+01	0.305275723761D+02	0.225361039807D+03	

rx (km)	ry (km)	rz (km)	rmag (km)
-.473889037681D+04	-.454354842152D+04	-.236806227854D+04	0.697915705635D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.569856258595D+01	-.831667145066D+01	-.462318936301D+01	0.110911910745D+02

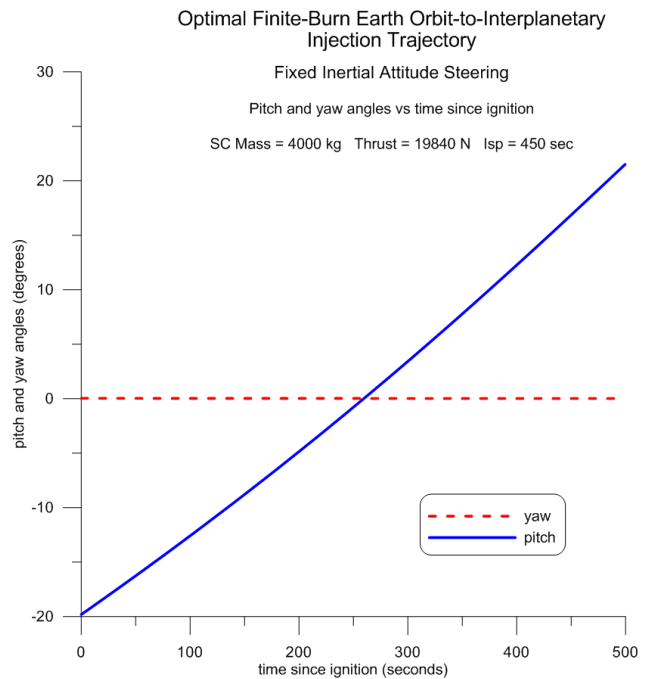
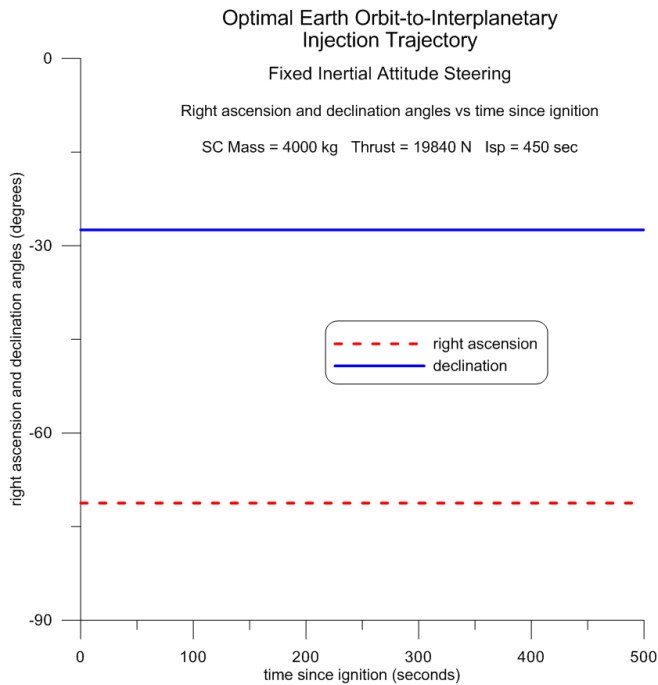
-----  
 outgoing hyperbola  
 -----

right ascension	349.680040000000	degrees
declination	-6.66625299999999	degrees
orbital energy	8.78856400000002	(km/sec) **2
final coast duration	100.000000000000	seconds
	1.66666666666667	minutes

-----  
 verification of OCS solution  
 -----

right ascension	349.680040437411	degrees
declination	-6.66625277125702	degrees
orbital energy	8.78856365891086	(km/sec) **2
final mass	1754.15735504558	kilograms
propellant mass	2245.84264495442	kilograms
delta-v	3637.65513253191	meters/second

The following are plots of the behavior of the inertial right ascension and declination angles, and the yaw and pitch angles during the propulsive maneuver.



## Verification of the optimal control solution

The optimal control solution determined by the *AMA\_OC* software can be verified by numerically integrating the orbital equations of motion with the OC-computed initial park orbit conditions and the optimal control solution. This is equivalent to solving an initial value problem (IVP) that uses the OC optimal unit thrust vector solution. This part of the *hyper\_ocs* computer program uses a Runge-Kutta-Fehlberg 7(8) variable step size method to integrate the orbital equations of motion.

The following is a display of the final solution computed using this *explicit* numerical integration method. These results are for the variable attitude example.

```
-----  
verification of optimal control solution  
-----  
  
right ascension      349.680039311870      degrees  
declination          -6.66625334384853      degrees  
orbital energy        8.78856386226052      (km/sec)**2  
  
final mass           1764.41952921893      kilograms  
propellant mass      2235.58047078107      kilograms  
delta-v              3611.91346050931      meters/second
```

A comparison of these results with the earlier OC solution exhibits excellent agreement. The delta-v was computed by including the thrust acceleration in the verification equations of motion.

## Creating an Initial Guess

This section describes several methods that can be used to create an initial guess for *hyper\_ocs*. The *AMA\_OC* software attempts to obtain problem feasibility before trying to solve the optimal control problem. If the *AMA\_OC* algorithm cannot find a feasible solution, the software will print warnings and the user will have to provide a better initial guess.

The software requires an initial guess for the thrust duration. The user should also provide lower and upper bounds for the total thrust duration. These three inputs should be in seconds.

For hyperbolic injection from circular and near-circular orbits, the delta-v magnitude can be estimated from the *Impulsive Interplanetary Injection from a Circular Earth Orbit* algorithm described in Appendix B. This algorithm calculates the *impulsive* delta-v required to perform the injection.

From the user-defined initial spacecraft mass and delta-v estimate, the propellant mass required for the maneuver is given by the ideal rocket equation as

$$m_p = m_i \left( 1 - e^{\frac{-\Delta V}{V_{ex}}} \right)$$

An estimate of the thrust duration of the maneuver can be determined from

$$t_d = \frac{g I_{sp} m_p}{F} = \frac{V_{ex} m_p}{F}$$

where

$F$  = thrust (newtons)

$m_i$  = pre-maneuver spacecraft mass (kilograms)

$V_{ex}$  = exhaust velocity =  $g I_{sp}$  (meters/second)

$I_{sp}$  = specific impulse (seconds)

$g$  = acceleration of gravity at Earth surface (meters/second<sup>2</sup>)

Since a finite burn maneuver will require a thrust duration longer than this impulsive estimate, the user can increase this value by about 10% and use it for the initial guess required by `hyper_ocs`.

For the numerical integration initial guess option with variable attitude steering, the software models the maneuver using tangential thrusting. For this case, the unit thrust vector in the modified equinoctial frame at all times is simply  $\mathbf{u}_T = [0 \ 1 \ 0]^T$ . Please note that this type of steering method creates a *coplanar* initial guess that increases the spacecraft's orbital energy.

For the fixed inertial attitude steering option, the software uses the user-defined initial guess for right ascension and declination to create a unit thrust vector in the Earth-centered-inertial (ECI) coordinate system. The software then transforms this inertial unit thrust vector to the modified equinoctial frame, and integrates the equations of motion in this system. For many simulations, the initial guesses for both angles can be set to zero. If the simulation does not converge with these simple guesses, the user can input "averaged" angles determined from the output of an optimized variable attitude case.

The dynamic variables at each grid point of the initial guess are determined by setting the initial guess option `INIT(1) = 6` with `INIT(2) = 2` within the `odeinp` subroutine for this aerospace trajectory optimization problem. These program options create an initial guess from the numerical integration of the equations of motion coded in the `oderhs` subroutine. The `INIT(1) = 6` program option tells the `AMA_OC` software to construct an initial guess by solving an initial value problem (IVP) with a linear control approximation. The `INIT(2) = 2` program option tells the program to use the Dormand-Prince variable step size numerical method to solve the initial value problem.

Binary restart data files can also be used to initialize a `hyper_ocs` simulation. A typical scenario is

1. Create a binary restart file from a converged and optimized simulation
2. Modify the original input file with slightly different spacecraft characteristics, propulsive parameters or perhaps final mission targets and/or constraints
3. Use the previously created binary restart file as the initial guess for the new simulation

This techniques works well provided the two simulations are not dramatically different. Sometimes it is necessary to make successive small changes in the mission definition and run multiples simulations to eventually reach the final desired solution.

## Problem Setup

This part of the user's manual provides details about the software implementation within `hyper_ocs`. It defines such things as point and path constraints (boundary conditions), bounds on the dynamic variables, and the performance index or objective function.

### *(1) Point functions – initial orbit constraints*

The software allows the user to select one of the following initial orbit constraint options:

- 1) constrain semimajor axis, eccentricity and inclination
- 2) constrain all initial orbital elements
- 3) option 2 with unconstrained true longitude

For all three options, the initial orbit inclination is constrained by enforcing

$$\sqrt{h^2 + k^2} = \tan\left(\frac{i}{2}\right)$$

where  $i$  is the park orbit inclination. Furthermore, the semiparameter is constrained to the initial value according to

$$p_L = p_U = p_i$$

If the park orbit is circular, the software enforces the following two constraints:

$$f = 0 \quad g = 0$$

Otherwise, for an elliptical park orbit, the single constraint

$$\sqrt{f^2 + g^2} = e$$

is enforced, where  $e$  is the user-defined park orbit eccentricity.

For constraint option 2, both lower and upper bounds for the other modified equinoctial elements are set equal to the initial elements as follows:

$$f_L = f_U = f_i \quad g_L = g_U = g_i$$

$$h_L = h_U = h_i \quad k_L = k_U = k_i$$

$$L_L = L_U = L_i$$

Option 3 is identical to option 2 with the initial true longitude bounded according to

$$-180^\circ \leq L_i \leq +180^\circ$$

In optimal control terminology, these constraints or boundary conditions are called *point functions*.

(2) *Performance index – maximize final spacecraft mass*

The objective function or performance index  $J$  for this simulation is the mass of the spacecraft at burnout of the interplanetary injection stage. This is simply

$$J = m_f$$

The value of the `maxmin` indicator in the `AMA_OC` software tells the software whether the user is minimizing or maximizing the performance index. Since this simulation involves a single continuous maneuver, this is equivalent to minimizing the required propellant mass. The spacecraft mass at the initial time is constrained to the user-defined initial value.

Please note that the `hyper_ocs` software will not be able to optimize the propulsive maneuver using the fixed attitude steering option and a highly constrained initial park orbit. For this situation, the software will display a warning and internally set `maxmin` to zero.

(3) *Path constraint – unit thrust vector scalar magnitude*

For the *variable steering* program option, the scalar magnitude of the components of the unit thrust vector at any time during the simulation is constrained as follows:

$$|\mathbf{u}_T| = \sqrt{u_{T_x}^2 + u_{T_y}^2 + u_{T_z}^2} = 1$$

(4) *Point functions – final “scaled” asymptote unit vector*

At the final time the solution should satisfy the following “energy scaled” unit asymptote *targeted minus predicted* vector constraint

$$(C_3 \hat{\mathbf{s}})_T - (C_3 \hat{\mathbf{s}})_P = 0$$

where the user-defined “target” departure asymptote unit vector is given by

$$\hat{\mathbf{s}}_T = \begin{Bmatrix} \cos \delta_\infty \cos \alpha_\infty \\ \cos \delta_\infty \sin \alpha_\infty \\ \sin \delta_\infty \end{Bmatrix}$$

In this expression,  $\alpha_\infty$  is the user-defined RLA and  $\delta_\infty$  is the user-defined DLA of the departure orbit.

The predicted “twice specific” (per unit mass) orbital energy ( $C_3$ ) is determined from the final position  $\mathbf{r}$  and velocity  $\mathbf{v}$  vectors according to

$$C_3 = |\mathbf{v}|^2 - \frac{2\mu}{|\mathbf{r}|}$$

The predicted asymptote unit vector at any trajectory time can be computed from

$$\hat{\mathbf{s}}_p = \frac{1}{1 + C_3 \frac{h^2}{\mu^2}} \left\{ \left( \frac{\sqrt{C_3}}{\mu} \right) \mathbf{h} \times \mathbf{e} - \mathbf{e} \right\} = \frac{1}{1 + C_3 \frac{p}{\mu}} \left\{ \left( \frac{\sqrt{C_3}}{\mu} \right) \mathbf{h} \times \mathbf{e} - \mathbf{e} \right\}$$

where  $\mathbf{h}$  and  $\mathbf{e}$  are the predicted final angular momentum and orbital eccentricity vectors, respectively.

These vectors are computed using the following two equations;

$$\mathbf{h} = \mathbf{r} \times \mathbf{v}$$

$$\mathbf{e} = \frac{\mathbf{v} \times \mathbf{h}}{\mu} - \frac{\mathbf{r}}{r} = \frac{1}{\mu} \left[ \left( v^2 - \frac{\mu}{r} \right) \mathbf{r} - (\mathbf{r} \cdot \mathbf{v}) \mathbf{v} \right]$$

### *Bounds on the dynamic variables*

The following lower and upper bounds are applied to the spacecraft mass and the modified equinoctial dynamic variables *during* the propulsive maneuver.

$$0.05m_{sc_i} \leq m_{sc} \leq 1.05m_{sc_i}$$

$$p \geq 0.8p_i$$

$$-1 \leq h \leq +1$$

$$-1 \leq k \leq +1$$

where  $m_{sc_i}$  is the initial spacecraft mass. The elements  $f$  and  $g$  are unconstrained.

For the variable steering option, the radial, tangential and normal components of the unit thrust vector are constrained as follows:

$$-1.1 \leq u_r \leq +1.1$$

$$-1.1 \leq u_t \leq +1.1$$

$$-1.1 \leq u_n \leq +1.1$$

For the inertially fixed attitude steering option, the right ascension and declination angles are bounded as follows:

$$-180^\circ \leq \alpha \leq +180^\circ$$

$$-90^\circ \leq \delta \leq +90^\circ$$

### Technical Discussion

The modified equinoctial orbital elements are a set of orbital elements that are useful for trajectory analysis and optimization. They are valid for circular, elliptic, and hyperbolic orbits. These equations exhibit no singularity for zero eccentricity and orbital inclinations equal to 0 and 90 degrees. However,

two components of the orbital element set are singular for an orbital inclination of 180 degrees. For this application the equations are well-behaved as the spacecraft transitions from a circular or elliptical park orbit to hyperbolic flight conditions.

The relationship between direct modified equinoctial and classical orbital elements is defined by the following definitions

$$\begin{aligned} p &= a(1 - e^2) & f &= e \cos(\omega + \Omega) \\ g &= e \sin(\omega + \Omega) & h &= \tan(i/2) \cos \Omega \\ k &= \tan(i/2) \sin \Omega & L &= \Omega + \omega + \theta \end{aligned}$$

where

$$\begin{aligned} p &= \text{semiparameter} \\ a &= \text{semimajor axis} \\ e &= \text{orbital eccentricity} \\ i &= \text{orbital inclination} \\ \omega &= \text{argument of periapsis} \\ \Omega &= \text{right ascension of the ascending node} \\ \theta &= \text{true anomaly} \\ L &= \text{true longitude} \end{aligned}$$

The relationship between classical and modified equinoctial orbital elements is:

$$\begin{aligned} \text{semimajor axis} & \quad a = \frac{p}{1 - f^2 - g^2} \\ \text{orbital eccentricity} & \quad e = \sqrt{f^2 + g^2} \\ \text{orbital inclination} & \quad i = 2 \tan^{-1} \left( \sqrt{h^2 + k^2} \right) \\ \text{argument of periapsis} & \quad \omega = \tan^{-1} (g/f) - \tan^{-1} (k/h) \\ \text{right ascension of the ascending node} & \quad \Omega = \tan^{-1} (k/h) \\ \text{true anomaly} & \quad \theta = L - (\Omega + \omega) = L - \tan^{-1} (g/f) \end{aligned}$$

The mathematical relationships between an inertial state vector and the corresponding modified equinoctial elements are summarized as follows:

position vector

$$\mathbf{r} = \begin{bmatrix} \frac{r}{s^2} (\cos L + \alpha^2 \cos L + 2hk \sin L) \\ \frac{r}{s^2} (\sin L - \alpha^2 \sin L + 2hk \cos L) \\ \frac{2r}{s^2} (h \sin L - k \cos L) \end{bmatrix}$$

velocity vector

$$\mathbf{v} = \begin{bmatrix} -\frac{1}{s^2} \sqrt{\frac{\mu}{p}} (\sin L + \alpha^2 \sin L - 2hk \cos L + g - 2f hk + \alpha^2 g) \\ -\frac{1}{s^2} \sqrt{\frac{\mu}{p}} (-\cos L + \alpha^2 \cos L + 2hk \sin L - f + 2ghk + \alpha^2 f) \\ \frac{2}{s^2} \sqrt{\frac{\mu}{p}} (h \cos L + k \sin L + fh + gk) \end{bmatrix}$$

where

$$\alpha^2 = h^2 - k^2 \quad s^2 = 1 + h^2 + k^2$$

$$r = \frac{p}{w} \quad w = 1 + f \cos L + g \sin L$$

The system of first-order modified equinoctial equations of orbital motion are given by

$$\dot{p} = \frac{dp}{dt} = \frac{2p}{w} \sqrt{\frac{p}{\mu}} \Delta_t$$

$$\dot{f} = \frac{df}{dt} = \sqrt{\frac{p}{\mu}} \left[ \Delta_r \sin L + [(w+1) \cos L + f] \frac{\Delta_t}{w} - (h \sin L - k \cos L) \frac{g \Delta_n}{w} \right]$$

$$\dot{g} = \frac{dg}{dt} = \sqrt{\frac{p}{\mu}} \left[ -\Delta_r \cos L + [(w+1) \sin L + g] \frac{\Delta_t}{w} + (h \sin L - k \cos L) \frac{f \Delta_n}{w} \right]$$

$$\dot{h} = \frac{dh}{dt} = \sqrt{\frac{p}{\mu}} \frac{s^2 \Delta_n}{2w} \cos L$$

$$\dot{k} = \frac{dk}{dt} = \sqrt{\frac{p}{\mu}} \frac{s^2 \Delta_n}{2w} \sin L$$

$$\dot{L} = \frac{dL}{dt} = \sqrt{\mu p} \left( \frac{w}{p} \right)^2 + \frac{1}{w} \sqrt{\frac{p}{\mu}} (h \sin L - k \cos L) \Delta_n$$

where  $\Delta_r, \Delta_t, \Delta_n$  are *non-two-body* perturbations in the radial, tangential and normal directions, respectively. The radial direction is along the geocentric radius vector of the spacecraft measured positive in a direction away from the gravitational center, the tangential direction is perpendicular to this radius vector measured positive in the direction of orbital motion, and the normal direction is positive in the direction of the angular momentum vector of the spacecraft's orbit.

The equations of orbital motion can also be expressed in vector form as follows:

$$\dot{\mathbf{y}} = \frac{d\mathbf{y}}{dt} = \mathbf{A}(\mathbf{y})\mathbf{P} + \mathbf{b}$$

where

$$\mathbf{A} = \begin{pmatrix} 0 & \frac{2p}{w} \sqrt{\frac{p}{\mu}} & 0 \\ \sqrt{\frac{p}{\mu}} \sin L & \sqrt{\frac{p}{\mu}} \frac{1}{w} \{(w+1)\cos L + f\} & -\sqrt{\frac{p}{\mu}} \frac{g}{w} \{h \sin L - k \cos L\} \\ -\sqrt{\frac{p}{\mu}} \cos L & \sqrt{\frac{p}{\mu}} \frac{1}{w} \{(w+1)\sin L + g\} & \sqrt{\frac{p}{\mu}} \frac{f}{w} \{h \sin L - k \cos L\} \\ 0 & 0 & \sqrt{\frac{p}{\mu}} \frac{s^2 \cos L}{2w} \\ 0 & 0 & \sqrt{\frac{p}{\mu}} \frac{s^2 \sin L}{2w} \\ 0 & 0 & \sqrt{\frac{p}{\mu}} \frac{1}{w} \{h \sin L - k \cos L\} \end{pmatrix}$$

and

$$\mathbf{b} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \sqrt{\mu p} \left(\frac{w}{p}\right)^2 \end{bmatrix}^T$$

The total non-two-body acceleration vector is given by

$$\mathbf{P} = \Delta_r \hat{\mathbf{i}}_r + \Delta_t \hat{\mathbf{i}}_t + \Delta_n \hat{\mathbf{i}}_n$$

where  $\hat{\mathbf{i}}_r, \hat{\mathbf{i}}_t$  and  $\hat{\mathbf{i}}_n$  are unit vectors in the radial, tangential and normal directions. These unit vectors can be computed from the inertial position vector  $\mathbf{r}$  and velocity vector  $\mathbf{v}$  according to

$$\hat{\mathbf{i}}_r = \frac{\mathbf{r}}{|\mathbf{r}|} \quad \hat{\mathbf{i}}_n = \frac{\mathbf{r} \times \mathbf{v}}{|\mathbf{r} \times \mathbf{v}|} \quad \hat{\mathbf{i}}_t = \hat{\mathbf{i}}_n \times \hat{\mathbf{i}}_r = \frac{(\mathbf{r} \times \mathbf{v}) \times \mathbf{r}}{|\mathbf{r} \times \mathbf{v}| |\mathbf{r}|}$$

For *unperturbed* two-body motion,  $\mathbf{P} = 0$  and the first five equations of motion are simply  $\dot{p} = \dot{f} = \dot{g} = \dot{h} = \dot{k} = 0$ . Therefore, for two-body motion these modified equinoctial orbital elements are constant. The true longitude is often called the *fast variable* of this orbital element set.

### Non-spherical Earth Gravity

The non-spherical gravitational acceleration vector can be expressed as

$$\mathbf{g} = g_N \hat{\mathbf{i}}_N - g_r \hat{\mathbf{i}}_r$$

where

$$\hat{\mathbf{i}}_N = \frac{\hat{\mathbf{e}}_N - (\hat{\mathbf{e}}_N^T \hat{\mathbf{i}}_r) \hat{\mathbf{i}}_r}{\|\hat{\mathbf{e}}_N - (\hat{\mathbf{e}}_N^T \hat{\mathbf{i}}_r) \hat{\mathbf{i}}_r\|}$$

and

$$\hat{\mathbf{e}}_N = [0 \ 0 \ 1]^T$$

In these equations the north direction component is indicated by subscript  $N$  and the radial direction component is subscript  $r$ .

The contributions due to the *zonal* gravity effects of  $J_2, J_3, J_4$  are as follows:

$$g_N = -\frac{\mu \cos \phi}{r^2} \sum_{k=2}^4 \left(\frac{R_e}{r}\right)^k P_k' J_k$$

$$g_r = -\frac{\mu}{r^2} \sum_{k=2}^4 (k+1) \left(\frac{R_e}{r}\right)^k P_k J_k$$

where

- $\mu$  = gravitational constant
- $r$  = geocentric distance of the spacecraft
- $R_e$  = equatorial radius of the Earth
- $\phi$  = geocentric latitude
- $J_k$  = zonal gravity coefficient
- $P_k$  =  $k^{\text{th}}$  order Legendre polynomial

For a zonal only Earth gravity model, the east component is identically zero. Finally, the zonal gravity perturbation contribution is  $\mathbf{a}_g = \mathbf{Q}^T \mathbf{g}$ , where  $\mathbf{Q} = [\hat{\mathbf{i}}_r \ \hat{\mathbf{i}}_t \ \hat{\mathbf{i}}_n]$ .

For  $J_2$  effects only, the three components are as follows:

$$\Delta_{J_{2r}} = -\frac{3\mu J_2 R_e^2}{2r^4} \left[ 1 - \frac{12(h \sin L - k \cos L)^2}{(1+h^2+k^2)^2} \right]$$

$$\Delta_{J_{2t}} = -\frac{12\mu J_2 R_e^2}{r^4} \left[ \frac{(h \sin L - k \cos L)(h \cos L + k \sin L)}{(1+h^2+k^2)^2} \right]$$

$$\Delta_{J_{2n}} = -\frac{6\mu J_2 R_e^2}{r^4} \left[ \frac{(1-h^2-k^2)(h \sin L - k \cos L)}{(1+h^2+k^2)^2} \right]$$

### *Propulsive Thrust*

The acceleration due to propulsive thrust can be expressed as

$$\mathbf{a}_T = \frac{T}{m(t)} \hat{\mathbf{u}}_T(t)$$

where  $T$  is the constant thrust magnitude,  $m$  is the spacecraft mass and  $\hat{\mathbf{u}}_T = [u_{T_r} \quad u_{T_t} \quad u_{T_n}]^T$  is the unit pointing thrust vector expressed in the spacecraft-centered radial-tangential-normal coordinate system.

The propellant mass flow rate is determined from

$$\dot{m} = \frac{dm}{dt} = \frac{T}{g I_{sp}}$$

where  $g$  is the acceleration of gravity and  $I_{sp}$  is the specific impulse of the propulsive system. The product  $g I_{sp}$  is also called the *exhaust velocity*.

The spacecraft mass at any mission elapsed time  $t$  is given by  $m(t) = m_{sc_i} - \dot{m}t$  where  $m_{sc_i}$  is the initial mass of the spacecraft.

*For the variable steering option, the components of this unit vector are control variables.*

The components of the unit thrust vector can also be defined in terms of the in-plane pitch angle  $\theta$  and the out-of-plane yaw angle  $\psi$  as follows:

$$u_{T_r} = \sin \theta \quad u_{T_t} = \cos \theta \cos \psi \quad u_{T_n} = \cos \theta \sin \psi$$

Finally, the pitch and yaw angles can be determined from the components of the unit thrust vector according to

$$\theta = \sin^{-1}(u_{T_r})$$

$$\psi = \tan^{-1}(u_{T_n}, u_{T_t})$$

Both steering angles are defined with respect to a local-vertical, local-horizontal (LVLH) system located at the spacecraft. The in-plane pitch angle is positive above the “local horizontal” and the out-of-plane yaw angle is positive in the direction of the angular momentum vector. The inverse tangent calculation in the second equation is a four quadrant operation.

For either steering option, the software provides the steering angles and the components of the unit thrust vector in both the inertial and modified equinoctial coordinate systems. The following section summarizes the inertial-to/from-modified equinoctial coordinate transformations and the calculation of the inertial unit thrust vector in terms of right ascension and declination angles.

The relationship between a unit thrust vector in the ECI coordinate system  $\hat{\mathbf{u}}_{T_{ECI}}$  and the corresponding unit thrust vector in the modified equinoctial system  $\hat{\mathbf{u}}_{T_{MEE}}$  is given by

$$\hat{\mathbf{u}}_{T_{ECI}} = \begin{bmatrix} \hat{\mathbf{i}}_r & \hat{\mathbf{i}}_t & \hat{\mathbf{i}}_n \end{bmatrix} \hat{\mathbf{u}}_{T_{MEE}}$$

where

$$\hat{\mathbf{i}}_r = \frac{\mathbf{r}}{|\mathbf{r}|} = \hat{\mathbf{r}} \quad \hat{\mathbf{i}}_n = \frac{\mathbf{r} \times \mathbf{v}}{|\mathbf{r} \times \mathbf{v}|} = \hat{\mathbf{h}} \quad \hat{\mathbf{i}}_t = \hat{\mathbf{i}}_n \times \hat{\mathbf{i}}_r = \frac{(\mathbf{r} \times \mathbf{v}) \times \mathbf{r}}{|\mathbf{r} \times \mathbf{v}| |\mathbf{r}|}$$

This relationship can also be expressed as

$$\hat{\mathbf{u}}_{T_{ECI}} = [Q] \hat{\mathbf{u}}_{T_{MEE}} = \begin{bmatrix} \hat{\mathbf{r}}_x & (\hat{\mathbf{h}} \times \hat{\mathbf{r}})_x & \hat{\mathbf{h}}_x \\ \hat{\mathbf{r}}_y & (\hat{\mathbf{h}} \times \hat{\mathbf{r}})_y & \hat{\mathbf{h}}_y \\ \hat{\mathbf{r}}_z & (\hat{\mathbf{h}} \times \hat{\mathbf{r}})_z & \hat{\mathbf{h}}_z \end{bmatrix} \hat{\mathbf{u}}_{T_{MEE}}$$

In these equations,  $\mathbf{r}$  is the inertial position vector and  $\mathbf{v}$  is the inertial velocity vector of the spacecraft.

Finally, the transformation of the unit thrust vector in the ECI system to the modified equinoctial coordinate system is given by

$$\hat{\mathbf{u}}_{T_{MEE}} = [Q]^T \hat{\mathbf{u}}_{T_{ECI}}$$

In the `hyper_ocs` computer program, the components of the inertial unit thrust vector are defined in terms of the right ascension  $\alpha$  and the declination angle  $\delta$  as follows:

$$u_{T_{ECI_x}} = \cos \alpha \cos \delta \quad u_{T_{ECI_y}} = \sin \alpha \cos \delta \quad u_{T_{ECI_z}} = \sin \delta$$

Finally, the right ascension and declination angles can be determined from the components of the ECI unit thrust vector according to

$$\alpha = \tan^{-1}\left(u_{T_{ECl_y}}, u_{T_{ECl_x}}\right)$$

$$\delta = \sin^{-1}\left(u_{T_{ECl_z}}\right)$$

*The right ascension and declination angles are treated as problem parameters in the fixed inertial attitude steering option.*

## References and Bibliography

“On the Equinoctial Orbital Elements”, R. A. Brouke and P. J. Cefola, *Celestial Mechanics*, Vol. 5, pp. 303-310, 1972.

“A Set of Modified Equinoctial Orbital Elements”, M. J. H. Walker, B. Ireland and J. Owens, *Celestial Mechanics*, Vol. 36, pp. 409-419, 1985.

“Design of Lunar and Interplanetary Ascent Trajectories”, Victor C. Clarke, Jr., JPL Technical Report No. 32-30, March 15, 1962.

*An Introduction to the Mathematics and Methods of Astrodynamics*, Richard H. Battin, AIAA Education Series, 1987.

*Analytical Mechanics of Space Systems*, Hanspeter Schaub and John L. Junkins, AIAA Education Series, 2003.

*Spacecraft Mission Design*, Charles D. Brown, AIAA Education Series, 1992.

*Orbital Mechanics*, Vladimir A. Chobotov, AIAA Education Series, 2002.

“A Computer Simulation of the Orbital Launch Window Problem”, Archie C. Young and Pat R. Odom, AIAA 67-615, 1967.

“Launch Parameters for Interplanetary Flights”, W. C. Riddell, *American Rocket Society Journal*, December 1960.

“Survey of Numerical Methods for Trajectory Optimization”, John T. Betts, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 21, No. 2, March-April 1998.

“Using Sparse Nonlinear Programming to Compute Low Thrust Orbit Transfers”, John T. Betts, *The Journal of the Astronautical Sciences*, Vol. 41, No. 3, July-September 1993, pp. 349-371.

“Optimal Interplanetary Orbit Transfers by Direct Transcription”, John T. Betts, *The Journal of the Astronautical Sciences*, Vol. 42, No. 3, July-September 1994, pp. 247-268.

“Equinoctial Orbit Elements: Application to Optimal Transfer Problems”, Jean A. Kechichian, AIAA 90-2976, AIAA/AAS Astrodynamics Conference, Portland, OR, 20-22 August 1990.

“Optimal Low Thrust Trajectories to the Moon”, John T. Betts and Sven O. Erb, *SIAM Journal on Applied Dynamical Systems*, Vol. 2, No. 2, pp. 144-170, 2003.

# APPENDIX A

## Contents of the Simulation Summary and CSV Files

This appendix is a brief summary of the information contained in the simulation summary screen displays and CSV data file produced by the `hyper_ocs` software.

**sma (km)** = semimajor axis in kilometers  
**eccentricity** = orbital eccentricity (non-dimensional)  
**inclination (deg)** = orbital inclination in degrees  
**argper (deg)** = argument of perigee in degrees  
**raan (deg)** = right ascension of the ascending node in degrees  
**true anomaly (deg)** = true anomaly in degrees  
**arglat (deg)** = argument of latitude in degrees. The argument of latitude is the sum of true anomaly and argument of perigee.  
**period (min)** = orbital period in minutes. If the orbit is hyperbolic, the period is undefined and a value of 0 is displayed.  
**rx (km)** = x-component of the eci position vector in kilometers  
**ry (km)** = y-component of the eci position vector in kilometers  
**rz (km)** = z-component of the eci position vector in kilometers  
**rmag (km)** = geocentric position magnitude in kilometers  
**vx (kps)** = x-component of the eci velocity vector in kilometers/second  
**vy (kps)** = y-component of the eci velocity vector in kilometers/second  
**vz (kps)** = z-component of the eci velocity vector in kilometers/second  
**vmag (kps)** = velocity vector scalar magnitude in kilometers/seconds

The user-defined comma-separated-variable (CSV) disk file is created by the software and contains the following information:

**time (sec)** = time since ignition in seconds  
**time (min)** = time since ignition in minutes  
**semimajor axis (km)** = semimajor axis in kilometers  
**eccentricity** = orbital eccentricity (non-dimensional)  
**inclination (deg)** = orbital inclination in degrees  
**arg of perigee (deg)** = argument of perigee in degrees  
**raan (deg)** = right ascension of the ascending node in degrees  
**true anomaly (deg)** = true anomaly in degrees  
**period (min)** = orbital period in minutes  
**mass (kg)** = spacecraft mass in kilograms

**thracc (m/s\*\*2)** = thrust acceleration in meters per second squared  
**yaw (deg)** = thrust vector yaw angle in degrees  
**pitch (deg)** = thrust vector pitch angle in degrees  
**rasc (deg)** = thrust vector right ascension angle in degrees  
**decl (deg)** = thrust vector declination angle in degrees  
**perigee alt (km)** = perigee altitude in kilometers  
**apogee alt (km)** = apogee altitude in kilometers  
**ut-radial** = radial component of unit thrust vector  
**ut-tangential** = tangential component of unit thrust vector  
**ut-normal** = normal component of unit thrust vector  
**ut-eci-x** = x-component of the eci unit thrust vector  
**ut-eci-y** = y-component of the eci unit thrust vector  
**ut-eci-z** = z-component of the eci unit thrust vector  
**semi-parameter (km)** = orbital semiparameter in kilometers  
**f equinoctial element** =  $\text{ecc} * \cos(\text{argper} + \text{raan})$   
**g equinoctial element** =  $\text{ecc} * \sin(\text{argper} + \text{raan})$   
**h equinoctial element** =  $\tan(i/2) * \cos(\text{raan})$   
**k equinoctial element** =  $\tan(i/2) * \sin(\text{raan})$   
**true longitude (deg)** = orbital true longitude in degrees  
**rx (km)** = x-component of eci position vector in kilometers  
**ry (km)** = y-component of eci position vector in kilometers  
**rz (km)** = z-component of eci position vector in kilometers  
**rmag (km)** = geocentric radius magnitude in kilometers  
**vx (kps)** = x-component of eci velocity vector in kilometers per second  
**vy (kps)** = y-component of eci velocity vector in kilometers per second  
**vz (kps)** = z-component of eci velocity vector in kilometers per second  
**vmag (kps)** = scalar velocity vector in kilometers per second  
**fpa (deg)** = flight path angle in degrees  
**c33 (km/sec)\*\*2** = twice specific orbital energy  
**shat(1)** = x-component of c3-scaled unit asymptote vector  
**shat(2)** = y-component of c3-scaled unit asymptote vector  
**shat(3)** = z-component of c3-scaled unit asymptote vector  
**deltav (mps)** = accumulative delta-v in meters per second

## APPENDIX B

### Impulsive Interplanetary Injection from a Circular Earth Orbit

This appendix describes an algorithm that can be used to determine the trajectory characteristics of an interplanetary injection that uses a single impulsive maneuver to transfer a spacecraft from a circular park orbit to a departure hyperbola. The methodology is based on the equations derived in Chapter 4 of Richard Battin's classic text, *Astronautical Guidance*, and Chapter 11 of *An Introduction to the Mathematics and Methods of Astrodynamics*, also written by Professor Battin and published by the American Institute of Aeronautics and Astronautics (AIAA). This algorithm can be used to create a thrust duration initial guess for the `hyper_ocs` software.

The numerical algorithm is valid for geocentric orbit inclinations that satisfy the following geometric constraint

$$i \geq |\delta_\infty|$$

where  $i$  is the orbital inclination of the park orbit. The algorithm also assumes a circular Earth park orbit and that injection occurs impulsively at the perigee of the departure hyperbola. Whenever  $i > |\delta_\infty|$ , there will be two opportunities to establish a departure hyperbola that will satisfy the energy and orientation of the outgoing asymptote. One injection opportunity will occur while the spacecraft is ascending and the other while the spacecraft is descending along the park orbit. For the case where  $i = |\delta_\infty|$ , there will be a single injection opportunity.

#### *Orientation of the park orbit and departure hyperbola*

This section summarizes the equations used to determine the right ascension of the ascending node (RAAN) of the park orbit and the injection true anomaly on the park orbit.

A unit vector in the direction of the departure asymptote is given by

$$\hat{\mathbf{s}} = \begin{Bmatrix} \cos \delta_\infty \cos \alpha_\infty \\ \cos \delta_\infty \sin \alpha_\infty \\ \sin \delta_\infty \end{Bmatrix}$$

where

$\alpha_\infty$  = right ascension of departure asymptote

$\delta_\infty$  = declination of departure asymptote

The angle between the outgoing asymptote and the spin axis of the Earth is given by

$$\beta = \cos^{-1}(\hat{\mathbf{s}} \cdot \hat{\mathbf{z}})$$

where  $\hat{\mathbf{z}} = [0 \ 0 \ 1]^T$ . Note that  $\beta = 90^\circ - \delta_\infty$ .

The park orbit right ascension of the ascending node for each opportunity can be determined from

$$\Omega_1 = 180^\circ + \alpha_\infty + \sin^{-1}\left(\frac{\cot \beta}{\tan i}\right)$$

$$\Omega_2 = 360^\circ + \alpha_\infty - \sin^{-1}\left(\frac{\cot \beta}{\tan i}\right)$$

The true anomaly on the park orbit for each injection opportunity can be determined from

$$\theta_1 = \cos^{-1}\left(\frac{\cos \beta}{\sin i}\right) - \eta$$

$$\theta_2 = -\cos^{-1}\left(\frac{\cos \beta}{\sin i}\right) - \eta$$

where

$$\eta = \sin^{-1}\left(\frac{1}{1 + r_p V_\infty^2 / \mu}\right)$$

In the last equation,  $r_p$  is the geocentric radius of the park orbit and  $\mu$  is the gravitational constant of the Earth. The velocity vector at infinity  $V_\infty$  is determined from  $V_\infty = \sqrt{C_3}$ .

For a tangential impulsive injection maneuver that occurs at perigee of the hyperbola, the true anomaly on the hyperbola is zero. Furthermore, since the orbit transfer modeled by this software is coplanar, the right ascension of the ascending node computed above should be the same for both the park orbit and the launch hyperbola. This can be verified by examining the hyperbola's RAAN which is computed using the state vector at injection.

#### *Departure delta-V*

The velocity vector at any geocentric position vector  $\mathbf{r}$  required to achieve a launch hyperbola defined by  $V_\infty$ ,  $\alpha_\infty$  and  $\delta_\infty$  is given by

$$\mathbf{v}_h = \left(d + \frac{1}{2}V_\infty\right)\hat{\mathbf{s}} + \left(d - \frac{1}{2}V_\infty\right)\hat{\mathbf{r}}$$

where

$$d = \sqrt{\frac{\mu}{(1 + \cos \psi)r_p} + \frac{V_\infty^2}{4}}$$

and  $\psi$  is the angle between the spacecraft's position vector and the departure asymptote unit vector which can be computed using

$$\cos \psi = \hat{\mathbf{s}} \cdot \hat{\mathbf{r}}$$

The injection  $\Delta\mathbf{v}$  vector can be determined from  $\Delta\mathbf{v} = \mathbf{v}_h - \mathbf{v}_p$  where  $\mathbf{v}_p$  is the inertial velocity vector in the park orbit prior to injection and  $\hat{\mathbf{r}} = \mathbf{r}/|\mathbf{r}|$ . Finally, the scalar injection delta-v is  $\Delta v = |\Delta\mathbf{v}|$ .

This algorithm has been coded in a Windows compatible computer program named `hyper1` which can be downloaded at [www.cdeagle.com](http://www.cdeagle.com) in the *Interplanetary Mission Analysis* section.

### Example calculation

Here are the park orbit and departure hyperbola characteristics for a typical example.

- Park orbit altitude = 185.2 kilometers
- Park orbit inclination = 28.5 degrees
- C3 of the departure hyperbola = 9.28 km<sup>2</sup>/sec<sup>2</sup>
- Right ascension of outgoing asymptote = 352.59 degrees
- Declination of the outgoing asymptote = 2.27 degrees

The following is the `hyper1` program output for this example. We can see that there are two injection opportunities with different RAANs and injection true anomalies. However, the scalar magnitude of the injection delta-v is identical.

```

-----
Interplanetary Injection from a Circular Park Orbit
-----

departure hyperbola characteristics
-----
c3                      9.280000000000000      km**2/sec**2
asymptote right ascension 352.5900000000000      degrees
asymptote declination   2.270000000000000      degrees

orbital elements and state vector of park orbit at injection - opportunity #1
-----
      sma (km)      eccentricity      inclination (deg)      argper (deg)
0.6563340000D+04      0.0000000000D+00      0.2850000000D+02      0.0000000000D+00

      raan (deg)      true anomaly (deg)      arglat (deg)      period (min)
0.1767767337D+03      0.2507477991D+02      0.2507477991D+02      0.8819562703D+02

      rx (km)      ry (km)      rz (km)      rmag (km)
-.6072821513D+04      -.2106348521D+04      0.1327240269D+04      0.6563340000D+04

      vx (kps)      vy (kps)      vz (kps)      vmag (kps)
0.2948681176D+01      -.6379088912D+01      0.3368063861D+01      0.7793032157D+01

orbital elements and state vector of hyperbola at injection - opportunity #1
-----

```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.4295264009D+05	0.1152804111D+01	0.2850000000D+02	0.2507477991D+02
raan (deg)	true anomaly (deg)	arglat (deg)	
0.1767767337D+03	0.3600000000D+03	0.2507477991D+02	
rx (km)	ry (km)	rz (km)	rmag (km)
-.6072821513D+04	-.2106348521D+04	0.1327240269D+04	0.6563340000D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.4326433915D+01	-.9359678097D+01	0.4941770522D+01	0.1143427743D+02

injection delta-v vector and magnitude - opportunity #1

x-component of delta-v	1377.75273908206	meters/second
y-component of delta-v	-2980.58918468226	meters/second
z-component of delta-v	1573.70666162370	meters/second
delta-v magnitude	3641.24527527765	meters/second

orbital elements and state vector of park orbit at injection - opportunity #2

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.6563340000D+04	0.0000000000D+00	0.2850000000D+02	0.0000000000D+00
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.3484032663D+03	0.2145979019D+03	0.2145979019D+03	0.8819562703D+02
rx (km)	ry (km)	rz (km)	rmag (km)
-.5950748689D+04	-.2122224678D+04	-.1778253190D+04	0.6563340000D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.3201396036D+01	-.6411955694D+01	-.3060921069D+01	0.7793032157D+01

orbital elements and state vector of hyperbola at injection - opportunity #2

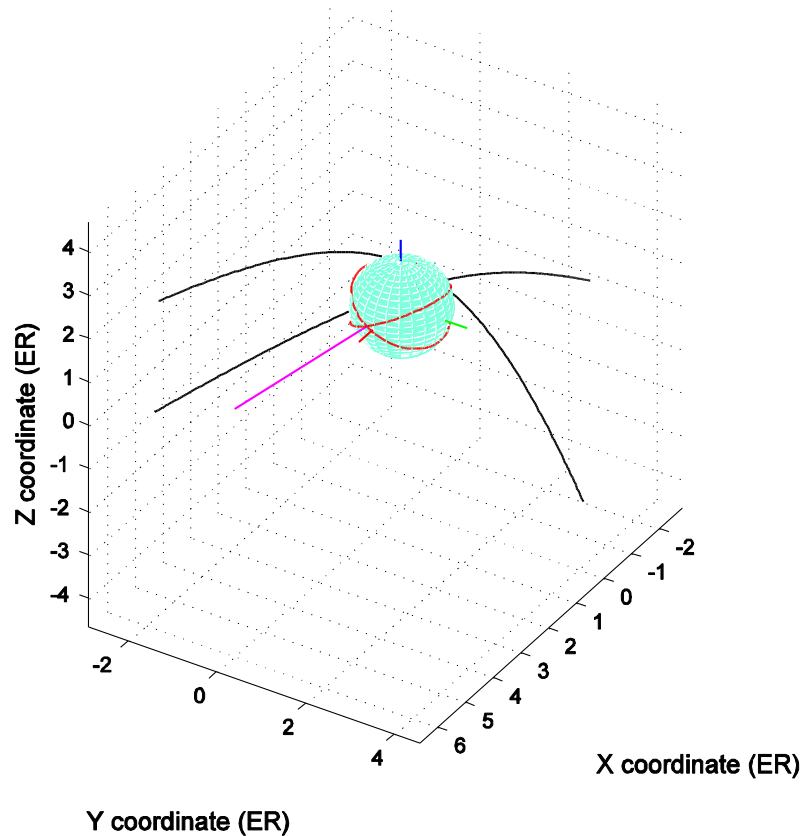
sma (km)	eccentricity	inclination (deg)	argper (deg)
-.4295264009D+05	0.1152804111D+01	0.2850000000D+02	0.2145979019D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.3484032663D+03	0.3600000000D+03	0.2145979019D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.5950748689D+04	-.2122224678D+04	-.1778253190D+04	0.6563340000D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.4697228205D+01	-.9407901676D+01	-.4491117193D+01	0.1143427743D+02

injection delta-v vector and magnitude - opportunity #2

x-component of delta-v	1495.83216868269	meters/second
y-component of delta-v	-2995.94598183915	meters/second

z-component of delta-v	-1430.19612353262	meters/second
delta-v magnitude	3641.24527527765	meters/second

The following is a graphics display illustrating the two injection opportunities for this example. This display is labeled with an Earth centered, inertial coordinate system. The x-axis of this system is red, the y-axis is green and the z-axis is blue. The outgoing asymptote is colored magenta, the park orbit traces are red, and the hyperbolic trajectories are black. Please note the units for each coordinate axis are Earth radii (ER).



### Finite-burn solutions

This section describes how to use `hyper_ocs` to optimize the finite-burn trajectories for each of the injection opportunities computed with the `hyper1` software.

For an initial spacecraft mass of 4000 kilograms, a specific impulse of 450 seconds, and a delta-v of 3641.245 meters/second, the propellant mass required for the impulsive maneuver is given by

$$m_p = m_i \left( 1 - e^{\frac{-\Delta V}{V_{ex}}} \right) = 4000 \cdot \left( 1 - e^{\frac{-3641.245}{9.80665 \cdot 450}} \right) = 2247.27 \text{ kilograms}$$

For a thrust level of 19840 newtons, the calculation for thrust duration is

$$t_d = \frac{g I_{sp} m_p}{F} = \frac{9.80665 \cdot 450 \cdot 2247.27}{19840} = 499.58 \text{ seconds}$$

With this answer for the thrust duration, we can create the following (partial) input data file for hyper\_ocs. Notice that we use a thrust duration initial guess of 550 seconds and the park orbit RAAN (176 degrees) of the first opportunity computed by hyper1. We also constrain only the semimajor axis, eccentricity and orbital inclination of the park orbit.

```

*****
** Optimal Finite-Burn Earth Orbit-to-Interplanetary Injection
** single phase, continuous-thrust maneuver with user-defined final coast
** variable and fixed inertial attitude steering
** input file ==> hyper2.in - February 14, 2011
*****

type of steering
1 = fixed inertial attitude
2 = variable attitude
-----
2

initial guess for right ascension (degrees)
0.0d0

initial guess for declination (degrees)
0.0d0

initial spacecraft mass (kilograms)
4000.0

thrust magnitude (newtons)
19840.0

specific impulse (seconds)
450.0

initial guess for thrust duration (seconds)
550.0

lower bound for thrust duration (seconds)
1.0

upper bound for thrust duration (seconds)
1000.0

*****
* INITIAL ORBIT *
*****

semimajor axis (kilometers)
6563.34

orbital eccentricity (non-dimensional)
0.0

orbital inclination (degrees)
28.5

argument of perigee (degrees)
0.0

right ascension of the ascending node (degrees)
176

true anomaly (degrees)
0.0

```

```

*****
initial orbit constraint options
*****
1 = constrain semimajor axis, eccentricity and inclination
2 = constrain all initial orbital elements
3 = option 2 with unconstrained true longitude
-----
1

*****
* LAUNCH HYPERBOLA *
*****

specific orbital energy (C3; [km/sec]**2)
9.28

right ascension of outgoing asymptote (RLA; degrees)
352.59

declination of outgoing asymptote (DLA; degrees)
2.27

final coast duration (seconds)
100.0

*****
* type of gravity model *
-----
1 = spherical Earth
2 = oblate Earth
-----
2

*****
* initial guess options *
*****
1 = numerical integration
2 = binary restart file
-----
1

```

Here's the hyper\_ocs solution for the first opportunity.

```

program hyper_ocs
=====

input file ==> hyper2.in

numerical integration initial guess

variable attitude steering

oblate earth gravity model

-----
beginning of finite burn
-----

      sma (km)      eccentricity      inclination (deg)      argper (deg)
0.656334000000D+04  0.127163366083D-16  0.285000000000D+02  0.000000000000D+00

      raan (deg)      true anomaly (deg)      arglat (deg)      period (min)
0.176807800508D+03  0.578903358972D+01  0.578903358972D+01  0.881956335064D+02

      rx (km)      ry (km)      rz (km)      rmag (km)
-.655213255829D+04  -.217269772647D+03  0.315887226893D+03  0.656334000000D+04

```

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.405405291239D+00	-.684692084631D+01	0.369954899383D+01	0.779303158492D+01

-----  
end of finite burn  
-----

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.429123585603D+05	0.115435598318D+01	0.284800431601D+02	0.251836707509D+02
raan (deg)	true anomaly (deg)	arglat (deg)	
0.176786768745D+03	0.208841694470D+02	0.460678401979D+02	
rx (km)	ry (km)	rz (km)	rmag (km)
-.499940493858D+04	-.407207922203D+04	0.235767389573D+04	0.686545738673D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.593950144357D+01	-.842217918403D+01	0.438127782797D+01	0.111984987839D+02

final mass	1741.10030901186	kilograms
propellant mass	2258.89969098814	kilograms
thrust duration	502.444929161942	seconds
	8.37408215269904	minutes
delta-v	3670.62646757410	meters/second

-----  
end of final coast  
-----

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.429526337823D+05	0.115421436692D+01	0.284776394025D+02	0.251931783309D+02
raan (deg)	true anomaly (deg)	arglat (deg)	
0.176780639937D+03	0.297298409658D+02	0.549230192967D+02	
rx (km)	ry (km)	rz (km)	rmag (km)
-.437687317644D+04	-.488843426872D+04	0.278088318892D+04	0.712650824125D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.648875162967D+01	-.789882809086D+01	0.408029172582D+01	0.110065509352D+02

-----  
outgoing hyperbola  
-----

right ascension	352.590000000000	degrees
declination	2.27000000000005	degrees
orbital energy	9.27999999999990	(km/sec)**2
final coast duration	100.000000000000	seconds
	1.66666666666667	minutes

-----  
verification of OCS solution  
-----

right ascension	352.590000136648	degrees
declination	2.26999992314201	degrees
orbital energy	9.27999983738958	(km/sec)**2

final mass	1741.10030901414	kilograms
propellant mass	2258.89969098586	kilograms
delta-v	3670.62604328145	meters/second

Here's the hyper\_ocs solution for the second opportunity. The only change to the input data file involved setting the RAAN of the park orbit to 348 degrees.

```
program hyper_ocs
=====
```

```
input file ==> hyper2.in
```

```
numerical integration initial guess
```

```
variable attitude steering
```

```
oblate earth gravity model
```

```
-----
beginning of finite burn
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.656334000000D+04	0.337784797333D-15	0.285000000000D+02	0.000000000000D+00
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.348438209294D+03	0.195324084717D+03	0.195324084717D+03	0.881956335064D+02
rx (km)	ry (km)	rz (km)	rmag (km)
-.650706632841D+04	-.224735623270D+03	-.827655515878D+03	0.656334000000D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.693905143549D+00	-.688391171243D+01	-.358630680098D+01	0.779303158492D+01

```
-----
end of finite burn
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.429180375328D+05	0.115433556073D+01	0.284786048967D+02	0.214713200321D+03
raan (deg)	true anomaly (deg)	arglat (deg)	
0.348406880900D+03	0.208981941128D+02	0.235611394434D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.479953648441D+04	-.409934688621D+04	-.270163893604D+04	0.686578825041D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.626270382532D+01	-.846323354961D+01	-.381467808151D+01	0.111981940591D+02

final mass	1741.03235436989	kilograms
propellant mass	2258.96764563011	kilograms
thrust duration	502.460044249510	seconds
	8.37433407082516	minutes
delta-v	3670.79881229368	meters/second

```
-----
end of final coast
-----
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.429526337823D+05	0.115421378015D+01	0.284764855902D+02	0.214723293631D+03

raan (deg)	true anomaly (deg)	arglat (deg)	
0.348399158345D+03	0.297436442833D+02	0.244466937914D+03	
rx (km)	ry (km)	rz (km)	rmag (km)
-.414594745792D+04	-.491964777121D+04	-.306625412665D+04	0.712697192963D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.678656978359D+01	-.793668058918D+01	-.347684588160D+01	0.110062203088D+02

-----  
outgoing hyperbola  
-----

right ascension	352.590000000000	degrees
declination	2.27000000000001	degrees
orbital energy	9.27999999999994	(km/sec)**2
final coast duration	100.000000000000	seconds
	1.66666666666667	minutes

-----  
verification of optimal control solution  
-----

right ascension	352.590000184595	degrees
declination	2.27000008928881	degrees
orbital energy	9.27999983772533	(km/sec)**2
final mass	1741.03235436945	kilograms
propellant mass	2258.96764563055	kilograms
delta-v	3670.79828447177	meters/second

The finite-burn solution requires a delta-v which is about 29 meters per second larger than the impulsive solution.

## APPENDIX C

### Typical AMA\_OC Configuration File

The `hyper_ocs` computer program can read and use a user-defined configuration file. A description of each element in this file can be found in the **INOCS** routine in section 6.2, *Subprograms for Optimal Control*, and the **INSNLP** routine in Section 2.2, *Subprograms for Optimization* of the `AMA_OC` user's manual. Please note that the `hyper_ocs` software can read and use a subset of the information in this file. For example, a subset configuration file might contain only the following information;

```
ODETOL=0.1D-06
INSNLP:IOFLAG=5
SOCOUT=I4K4
```

The following is a typical “full version” configuration file created during the execution of the `hyper_ocs` software.

```
AEQTOL=0.1000000000000000D-02
DTAUX=0.0000000000000000D+00
OBJCTL=0.1000000000000000D-04
ODETOL=0.1000000011686097D-06
PGDCTL=0.1000000000000000D-02
PRTMSD=0.1490116119384766D-07
PRTMXD=0.1000000000000000D-02
PRTSFD=0.1000000000000000D-04
QDRTOL=0.1000000000000000D-02
RESTOL=0.1000000000000000D-04
SMLTOL=0.1490116119384766D-10
TOLJSD=0.1000000000000000D-05
TOLM5A=0.1490116119384766D-07
TOLM5R=0.1490116119384766D-07
IDSCPH=0
IDSCND=0
IDSCVR=0
IDSCFN=0
IDTSFD=-1
IPFAUX=0
IPFSFD=0
IPRSFD=1
IPGRD=0
IPNLP=10
IPODE=0
IPUAUX=0
IPUOCP=6
IRSTRT=2
ISCALE=0
ISFHES=41
ISFINP=42
ISFRST=43
ISFSCL=44
ITSWCH=2
M5DTYP=0
MITODE=20
MTSWCH=-1
MXDATA=0
MXPARM=10
MXPCON=20
MXSTAT=20
MXTERM=50
NPTAUX=100
NSSWCH=-1
SOCOUT=A0B0C0D0E0F0G0H0I0J2K0L0M0N0O0P0Q0R0S1T0U0V0W0X0Y0Z0
SPRTHS=SPARSE
NLPALG=SNLPMN
NLPOMR=M
KEYDPL=.lueiLUE
RHSTMP=RHSTMPLT
```

RSTFIL=hyper1.rsbin  
SCLFIL=scalewgt.fil  
INSNLP:ALFLWR=0.000000000000000D+00  
INSNLP:ALFUPR=0.100000000000000D+01  
INSNLP:CONTOL=0.1490116119384766D-07  
INSNLP:EPSRLF=0.1490116119384766D-07  
INSNLP:OBJTOL=0.9999999747378752D-05  
INSNLP:PGDTOL=0.100000000000000D-04  
INSNLP:SLPTOL=0.900000000000000D+00  
INSNLP:SFZTOL=0.100000000000000D-01  
INSNLP:TOLFIL=0.200000000000000D+01  
INSNLP:TOLKTC=0.1110953834938985D+26  
INSNLP:TOLPVT=0.100000000000000D-02  
INSNLP:IHESHN=0  
INSNLP:IOFLAG=5  
INSNLP:IOFLIN=-1  
INSNLP:IOFMFR=0  
INSNLP:IOFPAT=0  
INSNLP:IOFSHR=0  
INSNLP:IOFSRC=0  
INSNLP:IPUDRF=0  
INSNLP:IPUFZF=0  
INSNLP:IPUMF1=11  
INSNLP:IPUMF2=12  
INSNLP:IPUMF3=13  
INSNLP:IPUMF4=14  
INSNLP:IPUMF5=15  
INSNLP:IPUMF6=16  
INSNLP:IPUMF7=17  
INSNLP:IPUNLP=6  
INSNLP:IPUSTF=0  
INSNLP:IRELAX=1  
INSNLP:ITDRQP=-1  
INSNLP:ITFZQP=-1  
INSNLP:IT1MAX=20  
INSNLP:JACPRM=0  
INSNLP:LYNFNC=0  
INSNLP:LYNOUT=0  
INSNLP:LYNPLT=0  
INSNLP:LYNPNT=101  
INSNLP:LYNVAR=0  
INSNLP:MAXLYN=5  
INSNLP:MAXNFE=50000  
INSNLP:MNSAME=2  
INSNLP:NEWTON=0  
INSNLP:NITMAX=1000  
INSNLP:NITMIN=0  
INSNLP:NORMAL=0  
INSNLP:ALGOPT=FM  
INSNLP:KTOPTN=SMALL  
INSNLP:QPOPTN=SPARSE  
INSNLP:BIGCON=-0.100000000000000D+01  
INSNLP:FEATOL=0.100000000000000D-01  
INSNLP:PMULWR=0.100000000000000D+00  
INSNLP:PTHOL=0.100000000000000D+02  
INSNLP:RHOLWR=0.100000000000000D+03  
INSNLP:IMAXMU=10  
INSNLP:MUCALC=3  
INSNLP:MXQPIT=1