

# Program `ilt_ocs`

## Low-Thrust Interplanetary Trajectory Optimization

This document is the user's manual for a Fortran computer program called `ilt_ocs` that uses optimal control software distributed by Applied Mathematical Analysis to solve the low-thrust interplanetary flyby and rendezvous trajectory optimization problems. The software attempts to either maximize the final spacecraft mass or minimize the transfer time. The type of optimization is selected by the user. The interplanetary trajectory is modeled as a patched-conic, n-body system.

The important features of this scientific simulation are as follows:

- maximum payload or minimum transfer time optimization
- low-thrust, *patched-n-body* interplanetary *flyby* and *rendezvous* trajectory modeling
- user-defined bounds for throttle setting
- modified equinoctial orbital element equations of motion
- chemical or solar electric propulsion (SEP) models
- JPL DE421 planetary ephemeris model

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.

The `ilt_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 problem and perform the optimization using a sparse nonlinear programming method. The 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:

```
ilt_ocs e2m1_max_payload.in
```

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

```
*****
*           program ilt_ocs           *
*                                     *
*   low-thrust interplanetary        *
*   trajectory optimization          *
*                                     *
*           March 22, 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 `ilt_ocs` software is “data-driven” by a user-created text file. The following is a typical input file used by this computer program. This example is a maximum payload Earth-to-Mars rendezvous trajectory with an initial launch  $C_3$  of  $4.625 \text{ km}^2/\text{sec}^2$ . 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 increase or decrease the number of lines reserved for each comment. 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. ASCII text input is not case sensitive but must be spelled correctly.

In the following discussion the actual input file contents are in *courier* font and all explanations are in *times italic* font. Please note that the fundamental time argument in this computer program is Barycentric Dynamical Time (TDB). Furthermore, the fundamental coordinate system is the Earth mean ecliptic and equinox of J2000.

*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.*

```
*****
** low-thrust interplanetary trajectory optimization
** patched-conic, n-body heliocentric motion
** input file for ilt_ocs - e2m1_max_payload.in
** Earth-to-Mars max payload - March 21, 2011
*****
```

*The first input is an integer that defines the type of trajectory optimization to simulate.*

```
type of optimization (1 = maximum payload, 2 = minimum transfer time)
1
```

*The next input is an integer that tells the software what type of trajectory to model.*

```
trajectory type (1 = flyby, 2 = rendezvous)
2
```

*The next input specifies what type of propulsion system to use during the simulation. Option 1 is a constant thrust system and option 2 is a solar-electric propulsion (SEP) system.*

```
propulsion type (1 = chemical, 2 = SEP)
1
```

*The next input is the initial launch energy. This number is also called C3 which is twice the specific (per unit mass) orbital energy.*

```
launch energy ([km/sec]**2)
4.625d0
```

*The next input is the initial spacecraft mass, in kilograms.*

```
initial spacecraft mass (kilograms)
1171.1
```

*The next two inputs define the thrust and specific impulse for a constant thrust, chemical propulsion system (propulsion type option 1).*

```
thrust magnitude (newtons)
0.16831
```

```
specific impulse (seconds)
3070.0
```

*The next series of inputs define the characteristics of a solar electric propulsion system (propulsion option 2). Please see the Technical Discussion section later in this document for additional information about these data items and coefficients.*

```
solar array power at 1 AU (kw)
5.0
```

```
solar array minimum and maximum power (kw)
0.649, 2.6
```

```
solar array power coefficients
1.1063, 0.1495, -0.299, -0.0432, 0.0
```

```
SEP thrust magnitude coefficients
-1.9137, 36.242, 0.0, 0.0, 0.0
```

```
SEP propellant flow rate coefficients
0.47556, 0.90209, 0.0, 0.0, 0.0
```

*The lower and upper bounds for the throttle setting are defined by the next two user inputs.*

```
lower bound for throttle setting (0 <= bound <= 1)
0.0
```

```
upper bound for throttle setting (0 <= bound <= 1)
1.0
```

*The software allows the user to specify an initial guess for the launch and arrival calendar dates and lower and upper bounds on the actual dates found during the optimization process. For any guess for*

launch time  $t_L$  and user-defined launch time lower and upper bounds  $\Delta t_l$  and  $\Delta t_u$ , the launch time  $t$  is constrained as follows:

$$t_L - \Delta t_l \leq t \leq t_L + \Delta t_u$$

Likewise, for any guess for arrival time  $t_A$  and user-defined arrival time bounds  $\Delta t_l$  and  $\Delta t_u$ , the arrival time  $t$  is constrained as follows:

$$t_A - \Delta t_l \leq t \leq t_A + \Delta t_u$$

For fixed launch and/or arrival times, the lower and upper bounds are set to 0.

The next three inputs are the user's initial guess for the launch calendar date. Be sure to include all four digits of the calendar year.

```
*****
* LAUNCH CONDITIONS *
*****

launch calendar date initial guess (month, day, year)
7, 10, 2005
```

The next two inputs are the lower and upper bounds for the launch calendar date search interval. These values should be input in days.

```
launch date search boundary (days)
-60, +60
```

The next program input is an integer that specifies the launch planet.

```
*****
* launch planet *
*****
1 = Mercury
2 = Venus
3 = Earth
4 = Mars
5 = Jupiter
6 = Saturn
7 = Uranus
8 = Neptune
9 = Pluto
-----
3
```

The next set of inputs defines the user's initial guess for the arrival calendar date, the arrival date search intervals and the arrival celestial body.

```
*****
* ARRIVAL CONDITIONS *
*****

arrival calendar date initial guess (month, day, year)
3, 4, 2006

arrival date search boundary (days)
-120, +240

*****
* arrival celestial body *
*****
```

```

1 = Mercury
2 = Venus
3 = Earth
4 = Mars
5 = Jupiter
6 = Saturn
7 = Uranus
8 = Neptune
9 = Pluto
0 = asteroid/comet
-----

```

4

*The next series of inputs include the name and classical orbital elements of a comet or asteroid (arrival celestial body = 0). Please note that the angular orbital elements must be specified with respect to a heliocentric, Earth mean ecliptic and equinox of J2000 coordinate system.*

```

*****
* asteroid/comet orbital elements *
* (heliocentric, ecliptic J2000) *
*****

```

asteroid/comet name

**Tempel 1**

calendar date of perihelion passage (month, day, year)

**7, 5.3153, 2005**

perihelion distance (au)

**1.506167**

orbital eccentricity (nd)

**0.517491**

orbital inclination (degrees)

**10.5301**

argument of perihelion (degrees)

**178.8390**

longitude of the ascending node (degrees)

**68.9734**

*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 initial guess option 1 or an updated binary file. This feature is described in the next two sections.*

```

*****
* initial guess/restart option *
*****

```

1 = numerical integration

2 = binary data file

-----

1

*If the user elects to use a binary data file (option 2 above) for the initial guess, the following text input specifies the name of the file to use.*

name of initial guess/restart input data file

**e2m1\_max\_payload.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 options 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 restart 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 the AMA\_OC software. Options 2 and 3 allow the user to specify either the number of nodes (option 2) or time step size of the data file (option 3).

```
*****
* type of comma-delimited solution data file *
*****
1 = OC-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 (option 2) or the time step size of the data output in the solution file (option 3).

```
number of user-defined nodes or print step size in solution data file
1
```

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
e2m1_max_payload.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. For most simulations, the trapezoidal method is recommended.

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

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

The next input defines the relative error in the objective function. A value of 1.0d-5 is recommended.

```
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. A value of 1.0d-7 is recommended.*

```
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 NLP algorithm is controlled with the following integer input.*

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

*The level of output from the optimal control algorithm is controlled with the following integer input. Please note that option 4 will create lots of information.*

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

*The level of output from the differential equations algorithm is controlled with the following integer input. Please note that option 5 will create lots of information.*

```
*****
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 ilt\_ocs software. Please consult Appendix G.*

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

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

name of optimal control configuration file
e2m1_max_payload_config.txt

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

name of optimal control configuration file
e2m1_max_payload_config1.txt
```

## Optimal control solution and graphics

The ilt\_ocs software will create two comma-separated-variable (csv) output files. The first file contains the heliocentric, ecliptic state vector of the spacecraft and has the name specified by the user in the main input file. The second file (planets.csv) contains the state vectors of both celestial bodies. Please see Appendix A for additional information about the contents of these two files.

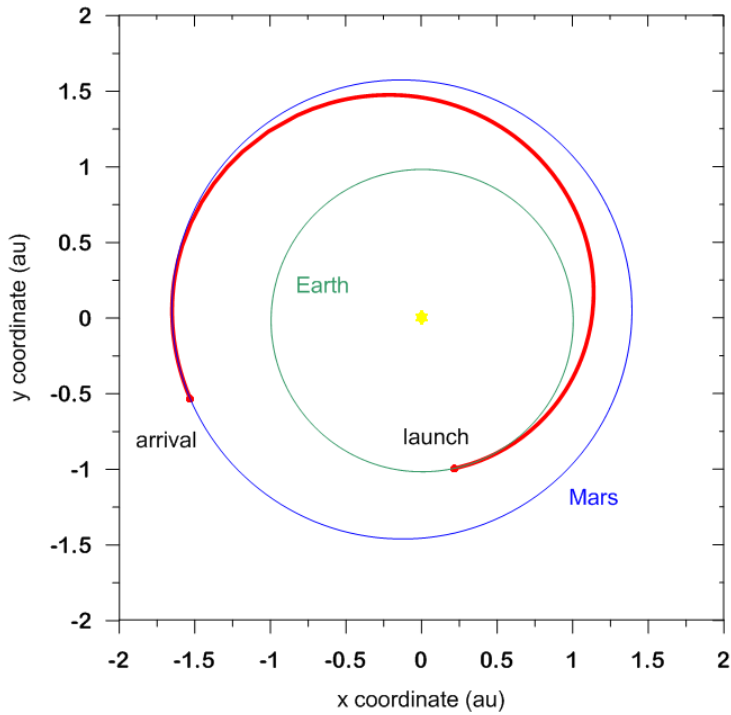
The software will also display a summary of the final solution and a numerical verification of the solution. This information is explained later in this document in the section titled Verification of the optimal control solution.

The following are typical transfer trajectory, optimal control and orbital element plots for this example. The first plot is a view of the trajectory and planetary orbits from the north pole of the ecliptic looking down on the ecliptic plane. The second plot illustrates the behavior of the pitch and yaw angles during the orbit transfer. The other plots illustrate the evolution of the heliocentric orbital elements, spacecraft mass, throttle setting and accumulated delta-v during the heliocentric orbital transfer.

It is interesting to note from these plots that although the simulation is initialized as a single phase, the AMA\_OC software is able to determine a multiple maneuver solution. Furthermore, it does this without a priori knowledge about the number of propulsive maneuvers or switching conditions.

# Low-thrust Interplanetary Trajectory Analysis Earth-to-Mars Maximum Final Mass

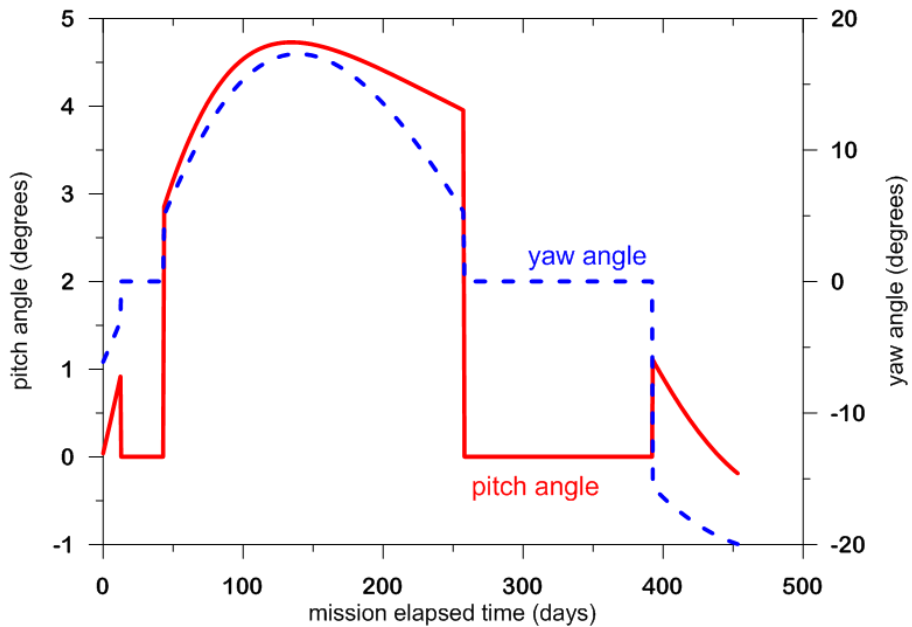
Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s



# Low-thrust Interplanetary Trajectory Analysis Earth-to-Mars Maximum Final Mass

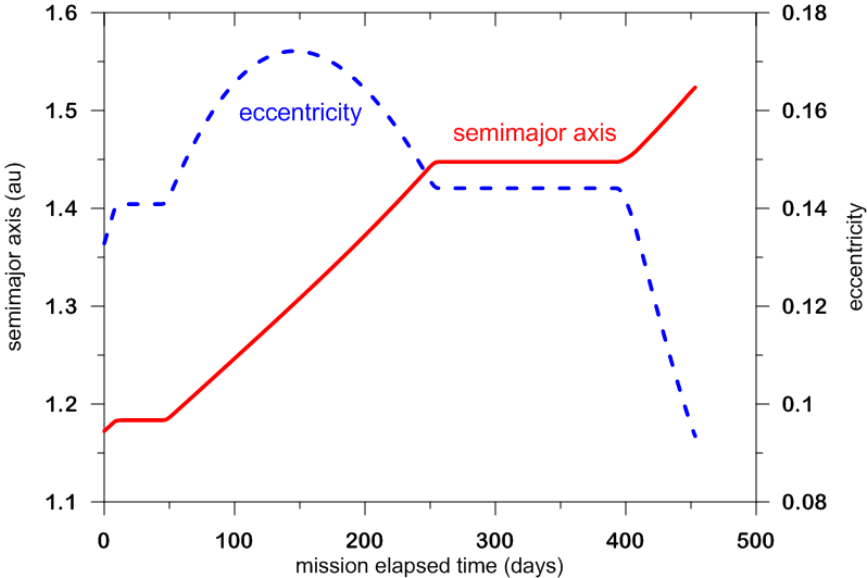
Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s

(optimal control angles vs. mission elapsed time)

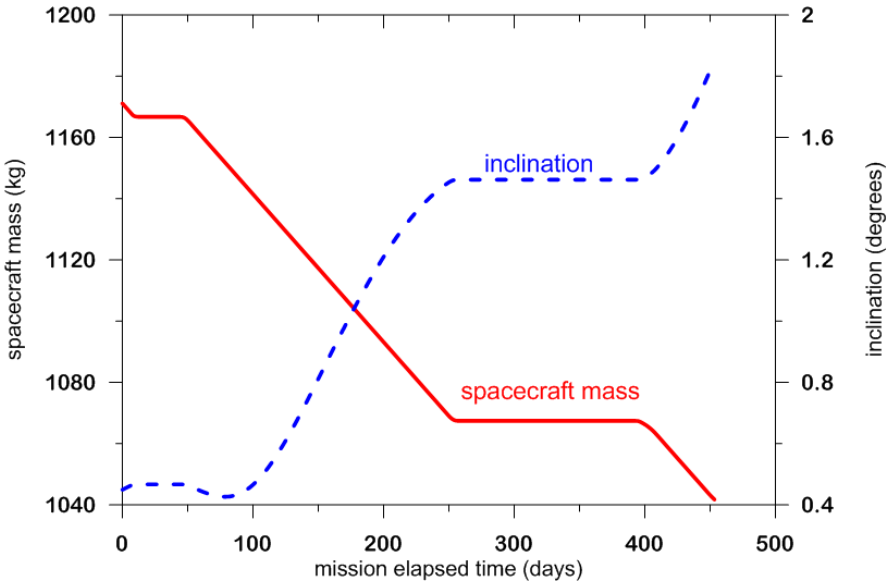


The next three plots are plots that illustrate the behavior of the semimajor axis, eccentricity, orbital inclination and throttle setting during the low-thrust orbital transfer.

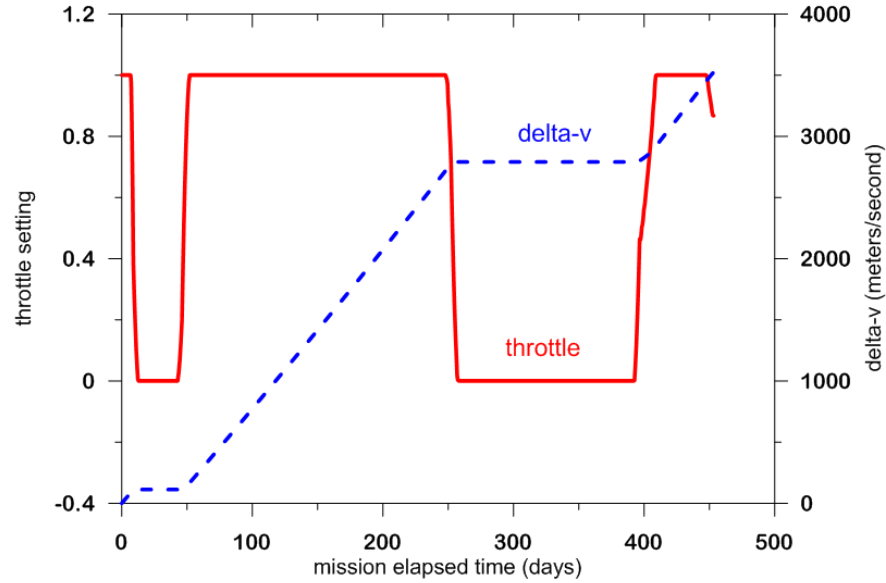
Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Maximum Final Mass  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
 (semimajor axis and eccentricity vs. mission elapsed time)



Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Maximum Final Mass  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
 (spacecraft mass and inclination vs. mission elapsed time)

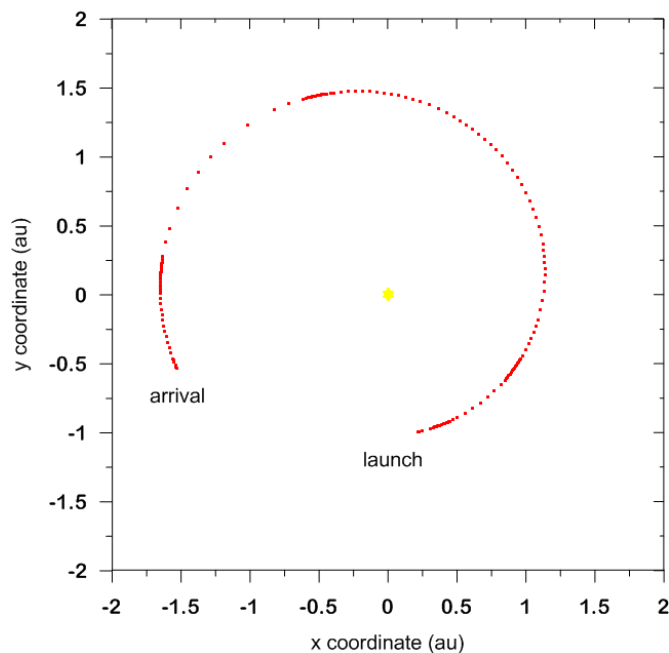


Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Maximum Final Mass  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
 (throttle and delta-v vs. mission elapsed time)



This plot illustrates how the mesh refinement feature of the AMA\_OC software has distributed the trajectory nodes of the optimal solution.

Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Maximum Final Mass  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s



## Verification of the optimal control solution

The optimal solution determined by the *AMA\_OC* software can be verified by numerically integrating the spacecraft's heliocentric equations of motion using the optimal control computed initial conditions and the optimal control determined by the *AMA\_OC* software. This computer program uses a Runge-Kutta-Fehlberg 7(8) variable step size method to explicitly integrate the orbital equations of motion.

The following is a typical display of the final solution and errors computed using the explicit numerical integration method. The errors are the differences between the spacecraft's final cartesian state vector and the arrival celestial body's ephemeris. The initial time is  $t_{zero}$  and the final time is  $t_{final}$ , both measured in days relative to the user-specified launch date initial guess.

```
program ilt_ocs
input file ==> e2m1_max_payload.in
rendezvous trajectory
maximum payload

LAUNCH CONDITIONS
-----

calendar date      07/04/2005
TDB time           11:12:04.672
TDB julian date    2453555.96672074

launch hyperbola characteristics
(Earth mean equator and equinox of J2000)
-----

right ascension    14.0052092520307      degrees
declination        -1.09827800615456      degrees
launch energy      4.62500000000000      (km/sec)**2

heliocentric orbital elements of the departure planet at launch
(Earth mean ecliptic and equinox of J2000)
-----

      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1000769973D+01  0.1595985178D-01  0.1346924405D-02  0.3369525119D+03

      raan (deg)    true anomaly (deg)      arglat (deg)      period (days)
0.1264776185D+03  0.1790720860D+03  0.1560245979D+03  0.3656788363D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.3292667100D+08  -.1484954302D+09  0.1452948353D+04  0.1521021317D+09

      vx (kps)      vy (kps)      vz (kps)      vmag (kps)
0.2860869963D+02  0.6335673965D+01  -.6293294032D-03  0.2930185078D+02
```

heliocentric orbital elements of the spacecraft at launch  
(Earth mean ecliptic and equinox of J2000)

-----

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1172507758D+01	0.1328504477D+00	0.4472981841D+00	0.1797865386D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.1025723223D+03	0.1433534199D+00	0.1799298920D+03	0.4637372842D+03
rx (km)	ry (km)	rz (km)	rmag (km)
0.3292667100D+08	-.1484954302D+09	0.1452948364D+04	0.1521021317D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.3069496859D+02	0.6796705358D+01	-.2454392343D+00	0.3143941063D+02

ARRIVAL CONDITIONS

-----

calendar date	09/30/2006	
TDB time	13:12:05.013	
TDB julian date	2454009.05005802	
flight time	453.083337281990	days
spacecraft mass	1041.74010208421	kilograms
delta-v	3523.98568030763	meters/second

heliocentric conditions of the destination celestial body at arrival  
(Earth mean ecliptic and equinox of J2000)

-----

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1523685751D+01	0.9343043699D-01	0.1849305087D+01	0.2865593844D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.4953837629D+02	0.2229171805D+03	0.1494765650D+03	0.6869759546D+03
rx (km)	ry (km)	rz (km)	rmag (km)
-.2292623909D+09	-.7906671107D+08	0.3975347044D+07	0.2425460618D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.8805666405D+01	-.2083594843D+02	-.6528878327D+00	0.2262968781D+02

heliocentric conditions of the spacecraft at arrival  
(Earth mean ecliptic and equinox of J2000)

-----

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1523685751D+01	0.9343043699D-01	0.1849305087D+01	0.2865593844D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.4953837629D+02	0.2229171805D+03	0.1494765650D+03	0.6869759546D+03
rx (km)	ry (km)	rz (km)	rmag (km)
-.2292623909D+09	-.7906671107D+08	0.3975347044D+07	0.2425460618D+09

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.8805666405D+01	-.2083594843D+02	-.6528878327D+00	0.2262968781D+02

INTEGRATED SOLUTION WITH OPTIMAL CONTROL  
 =====

tzero	-5.53327926122688	days
tfinal	447.550058020763	days

final heliocentric position vector and magnitude errors

delta rx	582.767748981714	kilometers
delta ry	-783.020385280252	kilometers
delta rz	-19.9336046096869	kilometers
delta rmag	976.287110234681	kilometers

final heliocentric velocity vector and magnitude errors

delta vx	8.919249560612741E-005	kilometers/second
delta vy	1.474770563802963E-005	kilometers/second
delta vz	-1.316307802645689E-006	kilometers/second
delta vmag	9.041310060088822E-005	kilometers/second
delta-v	3527.31138777129	meters/second

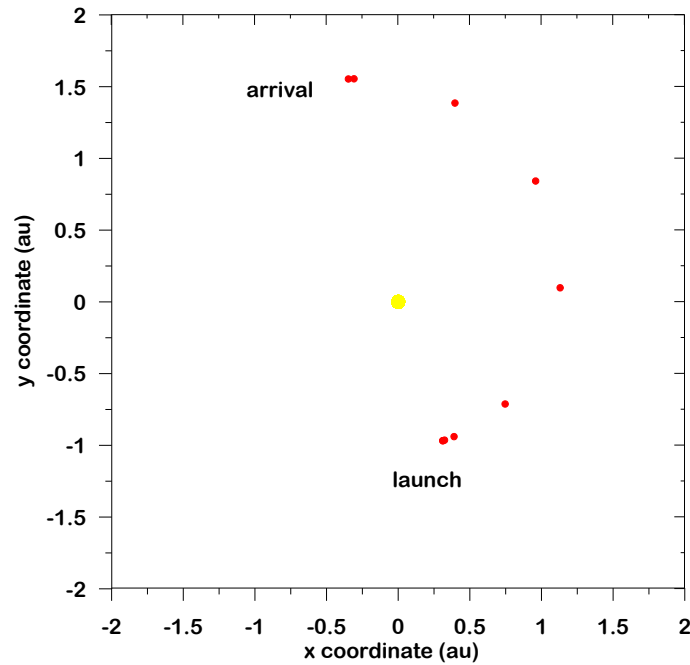
## Constructing an initial guess

An initial guess for the *AMA\_OC* algorithm is created by numerically integrating the modified equinoctial equations of motion for a user-defined orbital transfer time. During the interplanetary phase to an *outer planet*, the algorithm assumes *tangential thrusting* such that 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 approach creates a *coplanar* initial guess. However, since the orbital inclinations of most solar system objects of interest are relatively small, the software has little trouble finding a feasible trajectory and eventually an optimized solution. This initial guess algorithm uses the launch date, arrival date, propulsive characteristics and spacecraft mass provided by the user. For transfer to an *inner planet*, the unit thrust vector is  $\mathbf{u}_T = [0 \ -1 \ 0]^T$ . A throttle setting of one is used during the initial guess computations.

The dynamic variables and control 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. These program options create an initial guess from the numerical integration of the equations programmed 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 algorithm to use the Dormand-Prince variable step size numerical method to solve the initial value problem.

The following plot illustrates the location of typical trajectory grid points using this technique. This particular initial guess consists of ten grid points. These grid points are located at the integration steps determined by the Dormand-Prince algorithm.

**Low-thrust Interplanetary Trajectory Analysis**  
**Earth-to-Mars Maximum Final Mass**  
**Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s**



To model an impulsive delta-v at launch, the initial guess algorithm assumes that this maneuver is initially applied along the direction of the velocity vector of the departure planet. For outer planet missions, this direction is along the direction of orbital motion. For inner planet missions, this vector is aligned opposite to the direction of orbital motion.

The unit thrust vector of this maneuver in the heliocentric inertial coordinate system is

$$\mathbf{u}_T = \frac{\mathbf{r}_p}{|\mathbf{r}_p|}$$

where  $\mathbf{r}_p$  is the heliocentric inertial position vector of the departure planet at launch. Since the modified equinoctial form of the unit thrust vector is the optimal control in this example, the *AMA\_OC* software will change this unit pointing vector while searching for a solution.

An initial guess for the transfer time can be created by performing a one-dimensional minimization while numerically integrating the equations of motion with tangential thrusting. This process determines future close approach conditions between the spacecraft in its *coplanar* transfer orbit and the destination celestial body. This computational process can be performed by a utility computer program called *ca\_sc2body* that uses a minimization algorithm with the following objective function

$$f(t) = \Delta r(t) = |\mathbf{r}_p - \mathbf{r}_{sc}|$$

where  $\mathbf{r}_p$  is the heliocentric position vector of the arrival body and  $\mathbf{r}_{sc}$  is the heliocentric position vector of the spacecraft at any simulation time  $t$ . If the separation distance  $\Delta r$  is “close” enough (say 0.01 to perhaps 0.1 AU), the trajectory time provides a good initial guess. This close approach algorithm also uses the launch date, propulsive characteristics and spacecraft mass provided by the user. For consistency, this software also uses the same planetary ephemeris as the `ilt_ocs` computer program. Please see the `ca_sc2body` user’s manual for additional information. A Windows compatible executable version of the `ca_sc2body` computer program is available for download from the *Orbital and Celestial Mechanics Website* ([www.cdeagle.com](http://www.cdeagle.com)) in the Interplanetary Mission Analysis area.

## Problem setup

This section provides additional details about the software implementation. For good scaling the time unit used in all internal calculations is days, position is expressed in astronomical units and the velocity unit is astronomical units per day.

### (1) Launch and arrival time bounds

The software allows the user to specify an initial guess for the launch and arrival calendar dates and lower and upper bounds on the actual dates found during the optimization process. For any guess for launch time  $t_L$  and user-defined launch time lower and upper bounds  $\Delta t_l$  and  $\Delta t_u$ , the launch time  $t$  is constrained as follows:

$$t_L - \Delta t_l \leq t \leq t_L + \Delta t_u$$

Likewise, for any guess for arrival time  $t_A$  and user-defined arrival time bounds  $\Delta t_l$  and  $\Delta t_u$ , the arrival time  $t$  is constrained as follows:

$$t_A - \Delta t_l \leq t \leq t_A + \Delta t_u$$

For fixed launch and/or arrival times, the lower and upper bounds are set to 0.

### (2) Performance index

The objective function or performance index  $J$  for this simulation is either the final mass of the spacecraft or the total transfer time. This is simply

$$J = m_f \quad \text{or} \quad J = t_f$$

The value of the `maxmin` indicator tells the software whether the user is minimizing or maximizing the performance index. The spacecraft mass at the initial time is fixed to the initial value. However, the `ilt_ocs` software can be easily modified to make the initial mass the performance index.

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

At any point during the interplanetary transfer trajectory, the scalar magnitude of the components of the unit thrust vector is constrained as follows:

$$|\mathbf{u}_T| = \sqrt{u_{T_r}^2 + u_{T_t}^2 + u_{T_n}^2} = 1$$

This formulation avoids problems that may occur when using thrust steering angles as control variables.

(4) Point functions – position and velocity vector “matching” at launch

For any launch time  $t_L$  the optimal solution must satisfy the following state vector boundary conditions (equality constraints) at launch:

$$\begin{aligned}\mathbf{r}_{s/c}(t_L) - \mathbf{r}_p(t_L) &= 0 \\ \mathbf{v}_{s/c}(t_L) - \{\mathbf{v}_p(t_L) + \Delta\mathbf{v}(t_L)\} &= 0\end{aligned}$$

where  $\mathbf{r}_{s/c}$  and  $\mathbf{v}_{s/c}$  are the heliocentric, inertial position and velocity vectors of the spacecraft at the launch time  $t_L$ ,  $\mathbf{r}_p$  and  $\mathbf{v}_p$  are the heliocentric, inertial position and velocity vectors of the departure planet at the launch time, and  $\Delta\mathbf{v}$  is the user-specified available *impulsive* delta-v vector at launch. This delta-v might be provided by the launch vehicle or perhaps a high-thrust upper stage. The delta-v scalar magnitude is equal to the square root of the launch specific orbital energy, also called C3L.

In this constraint formulation, the delta-v contribution is modeled as  $\Delta\mathbf{v} = |\Delta\mathbf{v}|\hat{\mathbf{u}}_T$  where  $\hat{\mathbf{u}}_T$  is the unit thrust vector in the heliocentric, inertial coordinate system. Since the optimal control for this problem is the unit thrust vector in the modified equinoctial system,  $\hat{\mathbf{u}}_T$  is computed from

$$\hat{\mathbf{u}}_T = \begin{bmatrix} \hat{\mathbf{i}}_r & \hat{\mathbf{i}}_t & \hat{\mathbf{i}}_n \end{bmatrix} \hat{\mathbf{u}}_{mee}$$

where  $\hat{\mathbf{u}}_{mee}$  is the unit thrust vector in the modified equinoctial coordinate system and

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

(5) Point functions – position and velocity vector “matching” at arrival

For any arrival time  $t_A$  the optimal solution must satisfy the following state vector boundary conditions at arrival:

$$\begin{aligned}\mathbf{r}_{s/c}(t_A) - \mathbf{r}_p(t_A) &= 0 \\ \mathbf{v}_{s/c}(t_A) - \mathbf{v}_p(t_A) &= 0\end{aligned}$$

where  $\mathbf{r}_{s/c}$  and  $\mathbf{v}_{s/c}$  are the heliocentric, inertial position and velocity vectors of the spacecraft at the arrival time  $t_A$ , and  $\mathbf{r}_p$  and  $\mathbf{v}_p$  are the heliocentric, inertial position and velocity vectors of the destination planet at the arrival time. This system of launch and arrival state vector equality constraints ensures a rendezvous mission.

*For a flyby mission, the velocity vector point functions at arrival are not enforced.*

### **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 interplanetary orbital transfer.

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

$$0.5r_p \leq p \leq 2r_a$$

$$-1 \leq f \leq +1$$

$$-1 \leq g \leq +1$$

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

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

where  $r_p$  is the heliocentric periapsis radius of the initial orbit,  $r_a$  is the apoapsis radius of the final orbit, and  $m_{sc_i}$  is the initial spacecraft mass provided by the user. These two distances, in astronomical units, are determined from the orbital elements of the departure and arrival bodies at the user's initial guess for the corresponding calendar dates. Since we may allow the *AMA\_OC* software to change the actual launch time during the optimization, the true longitude is unconstrained.

The components of the unit thrust vector are bounded as follows:

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

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

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

Finally, the throttle setting is bounded to user-defined lower and upper bounds according to

$$\eta_L \leq \eta \leq \eta_U$$

where these bounds are typically between 0 and 1.

*The natural bounds for this trajectory formulation are ideal for numerical optimization.*

### **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. This feature is important for trajectories that may transition to and from 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.

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 summarized as follows:

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

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

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

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 the next six equations

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

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

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

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

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

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

where  $\Delta_r, \Delta_t, \Delta_n$  are *non-two-body* perturbations in the radial, tangential and normal directions, respectively. For an interplanetary spacecraft, the radial direction is along the heliocentric 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 along 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

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

and

$$\mathbf{b} = \left[ 0 \ 0 \ 0 \ 0 \ 0 \ \sqrt{\mu p} \left(\frac{w}{p}\right)^2 \right]^T$$

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.

The acceleration due to propulsive thrust can be expressed as

$$\mathbf{a}_T = \eta \frac{T}{m} \hat{\mathbf{u}}$$

where  $T$  is the thrust,  $m$  is the spacecraft mass,  $\hat{\mathbf{u}} = [u_r \quad u_t \quad u_n]$  is the unit pointing thrust vector expressed in the spacecraft-centered radial-tangential-normal coordinate system, and  $\eta$  is the throttle setting. 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:

$$\begin{aligned} u_r &= \sin \theta \\ u_t &= \cos \theta \cos \psi \\ u_n &= \cos \theta \sin \psi \end{aligned}$$

The pitch angle is positive above the “local horizontal” and the yaw angle is positive in the direction of the angular momentum vector.

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

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

This relationship can also be expressed as

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

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

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

For the case of tangential steering,

$$\hat{\mathbf{u}}_{T_{ECI}} = \left[ \left( \hat{\mathbf{h}} \times \hat{\mathbf{r}} \right)_x \quad \left( \hat{\mathbf{h}} \times \hat{\mathbf{r}} \right)_y \quad \left( \hat{\mathbf{h}} \times \hat{\mathbf{r}} \right)_z \right]^T$$

## Planetary Perturbations

The general vector equation for *point-mass* perturbations such as the Moon or planets is given by

$$\ddot{\mathbf{r}} = - \sum_{j=1}^n \mu_j \left[ \frac{\mathbf{d}_j}{d_j^3} + \frac{\mathbf{s}_j}{s_j^3} \right]$$

In this equation,  $\mathbf{s}_j$  is the vector from the primary body to the secondary body  $j$ ,  $\mu_j$  is the gravitational constant of the secondary body and  $\mathbf{d}_j = \mathbf{r} - \mathbf{s}_j$ , where  $\mathbf{r}$  is the position vector of the spacecraft relative to the primary body.

To avoid numerical problems, use is made of Richard Battin's  $F(q)$  function given by

$$F(q_k) = q_k \left[ \frac{3 + 3q_k + q_k^2}{1 + (\sqrt{1 + q_k})^3} \right]$$

where

$$q_k = \frac{\mathbf{r}^T (\mathbf{r} - 2\mathbf{s}_k)}{\mathbf{s}_k^T \mathbf{s}_k}$$

The acceleration due to other planets can now be expressed as

$$\ddot{\mathbf{r}} = - \sum_{k=1}^n \frac{\mu_k}{d_k^3} \left[ \mathbf{r} + F(q_k) \mathbf{s}_k \right]$$

Finally, the perturbation due to secondary bodies in the modified equinoctial coordinate system is given by

$$\mathbf{a} = [\mathbf{Q}]^T \mathbf{t}$$

where  $\mathbf{Q} = \left[ \hat{\mathbf{i}}_r \quad \hat{\mathbf{i}}_t \quad \hat{\mathbf{i}}_n \right]$ . All planetary point-mass perturbations except those due to the launch and arrival planets are included in the equations of motion.

## Planetary ephemeris

The software models the planetary coordinates using the DE 421 model from JPL. This planetary ephemeris provides position and velocity vectors relative to the Earth mean equator and equinox of J2000 (EME2000). The `ilt_ocs` software converts the DE 421 state vectors to the Earth mean ecliptic system using the following transformation

$$\mathbf{S}_{ec} = \begin{bmatrix} 1 & -0.000000479966 & 0 \\ 0.000000440360 & 0.917482137087 & 0.397776982902 \\ -0.000000190919 & -0.397776982902 & 0.917482137087 \end{bmatrix}^T \mathbf{S}_{eq}$$

where  $\mathbf{S}_{eq}$  is the state vector in the Earth mean equatorial frame, and  $\mathbf{S}_{ec}$  is the state vector in the Earth mean ecliptic frame.

### *Important Note*

The binary ephemeris file provided with this computer program was created for use on Windows compatible computers. For other platforms, you will need to create or obtain binary files specific to that system. Information and computer programs for creating these files can be found at the JPL solar system FTP site located at <ftp://ssd.jpl.nasa.gov/pub/eph/export/>. This site provides ASCII data files and Fortran computer programs for creating a binary file. A program for testing the user's ephemeris is also provided along with documentation.

## Asteroid and comet ephemeris

The orbital elements of an asteroid or comet relative to the mean ecliptic and equinox of J2000 coordinate system must be provided by the user. These elements can be obtained from the JPL Near Earth Object (NEO) website (<http://neo.jpl.nasa.gov>).

These orbital elements consist of the following items:

- TDB calendar date of perihelion passage
- perihelion distance (AU)
- orbital eccentricity (non-dimensional)
- orbital inclination (degrees)
- argument of perihelion (degrees)
- longitude of ascending node (degrees)

The software determines the mean anomaly of the asteroid or comet at any simulation time using the following equation:

$$M = \sqrt{\frac{\mu_s}{a^3}} t_{pp} = \sqrt{\frac{\mu_s}{a^3}} (JD - JD_{pp})$$

where  $\mu_s$  is the gravitational constant of the sun,  $a$  is the semimajor axis of the celestial body, and  $t_{pp}$  is the time since perihelion passage.

The semimajor axis is determined from the perihelion distance  $r_p$  and orbital eccentricity  $e$  according to

$$a = \frac{r_p}{(1 - e)}$$

This solution of Kepler's equation in this computer program is based on a numerical solution devised by Professor J.M.A. (Tony) Danby at North Carolina State University. Additional information about this algorithm can be found in "The Solution of Kepler's Equation", *Celestial Mechanics*, **31** (1983) 95-107, 317-328 and **40** (1987) 303-312.

The initial guess for Danby's method is

$$E_0 = M + 0.85 \text{sign}(\sin M) e$$

The fundamental transcendental equation we want to solve is

$$f(E) = E - e \sin E - M = 0$$

which has the first, second and third derivatives given by

$$f'(E) = 1 - e \cos E$$

$$f''(E) = e \sin E$$

$$f'''(E) = e \cos E$$

The iteration for an updated eccentric anomaly based on a current value  $E_n$  is given by the next four equations:

$$\Delta(E_n) = -\frac{f}{f'}$$

$$\Delta^*(E_n) = -\frac{f}{f' + \frac{1}{2} \Delta f''}$$

$$\Delta_n(E_n) = -\frac{f}{f' + \frac{1}{2} \Delta f'' + \frac{1}{6} \Delta^2 f'''}$$

$$E_{n+1} = E_n + \Delta_n$$

This algorithm provides quartic convergence of Kepler's equation. This process is repeated until the following convergence test involving the fundamental equation is satisfied:

$$|f(E)| \leq \varepsilon$$

where  $\varepsilon$  is the convergence tolerance. This tolerance is hardwired in the software to  $\varepsilon = 1.0e-10$ .

Finally, the true anomaly can be calculated with the following two equations

$$\sin \theta = \sqrt{1 - e^2} \sin E$$

$$\cos \theta = \cos E - e$$

and the four quadrant inverse tangent given by

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

If the orbit is hyperbolic, the initial guess is

$$H_0 = \log\left(\frac{2M}{e} + 1.8\right)$$

where  $H_0$  is the hyperbolic anomaly. The fundamental equation and first, second and third derivatives for this case are as follows:

$$f(H) = e \sinh H - H - M$$

$$f'(H) = e \cosh H - 1$$

$$f''(H) = e \sinh H$$

$$f'''(H) = e \cosh H$$

Otherwise, the iteration loop which calculates  $\Delta, \Delta^*$ , and so forth is the same. The true anomaly for hyperbolic orbits is determined with this next set of equations:

$$\sin \theta = \sqrt{e^2 - 1} \sinh H$$

$$\cos \theta = e - \cosh H$$

Finally, the true anomaly is determined from a four quadrant inverse tangent evaluation of these two equations.

### *Solar electric propulsion (SEP)*

For the SEP option, the power available, in kilowatts, for the engine is computed from

$$P = \frac{P_0}{r^2} \left( \frac{a_1 + \frac{a_2}{r} + \frac{a_3}{r^2}}{1 + a_4 r + a_5 r^2} \right)$$

where  $P_0$  is the power available at one astronomical unit (AU),  $r$  is the heliocentric distance of the spacecraft, in astronomical units, and the  $a_i$  's are coefficients provided by the user.

The propulsive thrust  $T$  and mass flow rate  $\dot{m}$  are computed from the following polynomials which are functions of the available power  $P$ .

$$T = c_1 + c_2P + c_3P^2 + c_4P^3 + c_5P^4$$

$$\dot{m} = d_1 + d_2P + d_3P^2 + d_4P^3 + d_5P^4$$

The coefficients for each polynomial are provided by the user. In these equations, the fundamental unit is milli-Newtons for thrust and milligrams for spacecraft mass. Finally, the software will also constrain the input power to the electric engine to the user-defined lower and upper bounds according to

$$P_L \leq P \leq P_U$$

### *Launch characteristics*

For interplanetary missions, the orientation of the departure hyperbola is specified in terms of the right ascension and declination of the outgoing asymptote. These coordinates can be calculated using the components of the initial spacecraft and departure planet heliocentric velocity vectors determined by the *AMA\_OC* software.

In this computer program, the heliocentric planetary coordinates and the transfer orbit velocity vectors are computed in the Earth mean ecliptic and equinox of J2000 coordinate system. In order to determine the orientation of the departure hyperbola in the EME2000 system, the initial heliocentric velocity vector must be transformed to the equatorial frame.

The required matrix-vector transformation is given by

$$\mathbf{V}_{\infty_{EQ}} = \begin{bmatrix} 1 & -0.000000479966 & 0 \\ 0.000000440360 & 0.917482137087 & 0.397776982902 \\ -0.000000190919 & -0.397776982902 & 0.917482137087 \end{bmatrix} \mathbf{V}_{\infty_{EC}}$$

where  $\mathbf{V}_{\infty_{EC}}$  is the v-infinity velocity vector in the ecliptic frame, and  $\mathbf{V}_{\infty_{EQ}}$  is the v-infinity velocity vector in the equatorial frame. The ecliptic v-infinity velocity vector can be computed using

$$\mathbf{V}_{\infty_{EC}} = \mathbf{V}_{sc} - \mathbf{V}_{dp}$$

where  $\mathbf{V}_{sc}$  is the heliocentric, ecliptic velocity vector of the spacecraft and  $\mathbf{V}_{dp}$  is the velocity vector of the departure planet, also in the ecliptic frame.

The right ascension of the asymptote is determined from

$$\alpha = \tan^{-1}(V_y, V_z)$$

and the geocentric declination of the outgoing asymptote is given by

$$\delta = 90^\circ - \cos^{-1}(\hat{V}_z)$$

where  $\hat{V}_z$  is the z-component of the unit v-infinity velocity vector in the equatorial frame of reference. The right ascension is computed using a four quadrant inverse tangent function.

## References and Bibliography

- “A Set of Modified Equinoctial Orbital Elements”, M. J. H. Walker, B. Ireland and J. Owens, *Celestial Mechanics*, Vol. 36, pp. 409-419, 1985.
- “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.
- “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.
- “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.
- “Modern Astrodynamics”, Victor R. Bond and Mark C. Allman, Princeton Univeristy Press, 1996.
- “Fuel-Optimal, Low-Thrust, Three-Dimensional Earth-Mars Trajectories”, R. S. Nah, S. R. Vadali, and E. Braden, *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 6, November-December 2001.
- “Optimal Low-Thrust Interception of Earth-Crossing Asteroids”, Bruce A. Conway, *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 5, September-October 1997.
- “Electric Propulsion Mission Analysis”, NASA SP-210, 1969.
- “Possibilities of Combining High- and Low-Thrust Engines in Flights to Mars”, G. G. Fedotov, *Cosmic Research*, Vol. 39, No. 6, 2001, pp. 613-621.
- “Design and Optimization of Interplanetary Spacecraft Trajectories”, Thomas T. McConaghy, PhD thesis, Purdue University, December 2004.
- “Interplanetary Mission Design Handbook, Volume 1, Part 2”, JPL Publication 82-43, September 15, 1983.
- “Error Analysis of Multiple Planet Trajectories”, F. M. Sturms, Jr., JPL Space Programs Summary, No. 37-27, Vol. IV.
- “The Planetary and Lunar Ephemeris DE 421”, W. M. Folkner, J. G. Williams, D. H. Boggs, JPL IOM 343R-08-003, 31-March-2008.
- “IERS Conventions (2003)”, IERS Technical Note 32, November 2003.
- “Planetary Constants and Models”, R. Vaughan, JPL D-12947, December 1995.

# 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 the CSV data files produced by the `ilt_ocs` software. All output is computed and displayed in a heliocentric, Earth mean ecliptic and equinox of J2000 coordinate system.

The simulation summary screen display contains the following information:

**calendar date** = calendar date of trajectory event  
**ephemeris time** = ephemeris time of trajectory event  
**julian date** = julian date of trajectory event  
**sma (au)** = semimajor axis in astronomical unit  
**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 (days)** = orbital period in days  
**delta-v** = scalar magnitude of the low-thrust maneuver in meters/seconds

The accumulated delta-v is computed from a cubic spline integration of the thrust acceleration at all grid points determined by the `AMA_OC` software.

The comma-separated-variable disk file is created by the `odeprt` subroutine and contains the following information:

**time (days)** = simulation time since launch in days  
**semimajor axis (au)** = semimajor axis in astronomical units  
**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  
**pitch** = thrust vector pitch angle in degrees  
**yaw** = thrust vector yaw angle in degrees  
**mass** = spacecraft mass in kilograms

**thracc** = thrust acceleration in meters/second\*\*2  
**rx (au)** = x-component of the spacecraft's heliocentric position vector in astronomical units  
**ry (au)** = y-component of the spacecraft's heliocentric position vector in astronomical units  
**rz (au)** = z-component of the spacecraft's heliocentric position vector in astronomical units  
**vx (km/sec)** = x-component of the spacecraft's heliocentric velocity vector in kilometers per second  
**vy (km/sec)** = y-component of the spacecraft's heliocentric velocity vector in kilometers per second  
**vz (km/sec)** = z-component of the spacecraft's heliocentric velocity vector in kilometers per second  
**ut-radial** = radial component of unit thrust vector  
**ut-tangential** = tangential component of unit thrust vector  
**ut-normal** = normal component of unit thrust vector  
**pmee** = orbital semiparameter  
**fmee** = modified equinoctial orbital element =  $\text{ecc} * \cos(\text{argper} + \text{raan})$   
**gmee** = modified equinoctial orbital element =  $\text{ecc} * \sin(\text{argper} + \text{raan})$   
**hmee** = modified equinoctial orbital element =  $\tan(i/2) * \cos(\text{raan})$   
**xkmee** = modified equinoctial orbital element =  $\tan(i/2) * \sin(\text{raan})$   
**xlmee** = true longitude in degrees  
**deltav (mps)** = accumulative delta-v in meters per second  
**throttle** = throttle setting  
**thrust** = propulsive thrust (Newtons)

The planets.csv file contains the following information:

**time (days)** = simulation time since launch in days  
**rp1-x (au)** = x-component of the launch planet heliocentric position vector in astronomical units  
**rp1-y (au)** = y-component of the launch planet heliocentric position vector in astronomical units  
**rp1-z (au)** = z-component of the launch planet heliocentric position vector in astronomical units  
**rp2-x (au)** = x-component of the destination body heliocentric position vector in astronomical units  
**rp2-y (au)** = y-component of the destination body heliocentric position vector in astronomical units  
**rp2-z (au)** = z-component of the destination body heliocentric position vector in astronomical units

# APPENDIX B

## Fortran Functions and Subroutines

This appendix is a brief summary of the major Fortran functions and subroutines included in the `ilt_ocs` computer program.

- `ilt_ocs.f` - main executive program
- `atan3.for` - four quadrant inverse tangent function
- `csint.for` - cubic spline integration of tabular data subroutine
- `eci2orb.for` - convert eci position and velocity vectors to classical orbital elements subroutine
- `gdate.for` - compute calendar date from Julian date subroutine
- `hci2mee.for` - convert heliocentric position and velocity vectors to modified equinoctial orbital elements subroutine
- `jd2str.for` - from a Julian date, print the character representation of calendar date and time subroutine
- `jpleph.for` - subroutine that reads and interpolates a JPL ephemeris file
- `julian.for` - subroutine to convert calendar date to Julian date
- `kepler1.for` - solve Kepler's equation using Danby's method subroutine
- `linput.for` - read and echo a line of text from an input file subroutine
- `mee2hci.for` - convert modified equinoctial orbital elements to heliocentric position and velocity vectors subroutine
- `meeeqms2.for` - modified equinoctial orbital elements equations of motion subroutine  
- used during the verification computations
- `odeinp.for` - simulation input subroutine
- `odepf.for` - point functions subroutine
- `odeprt.for` - print subroutine - creates comma-separated-variable file
- `oderhs.for` - subroutine that evaluates the equations of motion and any algebraic equations
- `oeprint.for` - subroutine that displays classical orbital elements
- `orb2eci.for` - convert classical orbital elements to position and velocity vectors subroutine
- `p2000.for` - subroutine that returns a planet's or asteroid/comet position and velocity vectors in kilometers and kilometers/second
- `readfpn.for` - read and echo floating point number from an input file subroutine
- `readint.for` - read and echo an integer from an input file subroutine

**readtext.for** - read and echo text from an input file subroutine  
**rkf78.for** - Runge-Fehlberg-Kutta (RKF78) numerical integration subroutine  
**rkf78cn.for** - evaluate RKF78 integration coefficients subroutine  
**utility.for** - number and text manipulation functions and subroutines  
**uvector.for** - unit vector subroutine  
**vcross.for** - vector cross product subroutine  
**vdot.for** - vector dot product subroutine  
**vecmag.for** - vector scalar magnitude function  
**xmod.for** - modulo 2 pi function

## APPENDIX C

### Example Fortran Subroutine

This appendix contains the source code for a single Fortran routine and illustrates typical programming conventions used in the `ilt_ocs` software. This subroutine is the point function routine required by the `AMA_OC` software.

```
      subroutine odepf(iphase, iphend, time, ydyn, nydyn, parm,
&                    nparm, ptf, nptf, iferr)

c     evaluate "position & velocity matching" point
c     functions at the beginning and end of phase 1

c     state variables

c     ydyn(1) = pmee = semiparameter of orbit
c     ydyn(2) = fmee = ecc * cos(argper + raan)
c     ydyn(3) = gmee = ecc * sin(argper + raan)
c     ydyn(4) = hmee = tan(i/2) * cos(raan)
c     ydyn(5) = xkmee = tan(i/2) * sin(raan)
c     ydyn(6) = xlmee = true longitude (radians)
c     ydyn(7) = spacecraft mass (kilograms)

c     control variables

c     ydyn(8) = radial component of unit thrust vector
c     ydyn(9) = tangential component of unit thrust vector
c     ydyn(10) = normal component of unit thrust vector
c     ydyn(11) = throttle setting

c     *****

      implicit double precision (a-h, o-z)

      include 'socxcom1.inc'

      parameter (zero = 0.0d0, one = 1.0d0, two = 2.0d0)

      parameter (nwork = 10)

      dimension work(nwork)

      dimension ydyn(nydyn), parm(nparm), ptf(nptf)

      dimension rsc(3), vsc(3), rp(3), vp(3)

      dimension utmee(3), uteci(3), qmat(3, 3)

      dimension xrdl(3), yrdl(3), zrdl(3)

c     unload modified equinoctial elements

      pmee = ydyn(1)

      fmee = ydyn(2)

      gmee = ydyn(3)

      hmee = ydyn(4)

      xkmee = ydyn(5)

      xlmee = ydyn(6)

c     current spacecraft mass
```

```

xmass = ydyn(7)

c  unload unit thrust vector in
c  modified equinoctial frame

utmee(1) = ydyn(8)

utmee(2) = ydyn(9)

utmee(3) = ydyn(10)

c  semiparameter error check

if (pmee .lt. zero) then
    iferr = -22
    return
endif

c  *****
c  convert modified equinoctial orbital elements
c  of spacecraft to heliocentric position and
c  velocity vectors (au and au/day)
c  *****

call mee2hci(pmee, fmee, gmee, hmee, xkmee, xlmee, rsc, vsc)

if (iphase .eq. 1 .and. iphend .eq. -1) then
c  *****
c  "position & velocity matching" at beginning of phase 1
c  *****

    xjdate = xjdateil + time

    call p2000(ip1, xjdate, rp, vp)

c  define position vector match point functions

    ptf(1) = rsc(1) - rp(1) / aunit
    ptf(2) = rsc(2) - rp(2) / aunit
    ptf(3) = rsc(3) - rp(3) / aunit

    if (vinfinity .ne. 0.0d0) then
c  compute radial frame unit vectors

        rmag = sqrt(rsc(1)**2 + rsc(2)**2 + rsc(3)**2)

        call dcopy(3, rsc, 1, xrdl, 1)

        call drscl(3, rmag, xrdl, 1)

        call vcross(rsc, vsc, zrdl)

        hmag = dnorm2(3, zrdl, 1)

        call drscl(3, hmag, zrdl, 1)

        call vcross(zrdl, xrdl, yrdl)

c  load elements of transformation matrix

        qmat(1, 1) = xrdl(1)
        qmat(1, 2) = yrdl(1)
        qmat(1, 3) = zrdl(1)

        qmat(2, 1) = xrdl(2)
        qmat(2, 2) = yrdl(2)
        qmat(2, 3) = zrdl(2)

```

```

    qmat(3, 1) = xrdl(3)
    qmat(3, 2) = yrdl(3)
    qmat(3, 3) = zrdl(3)

c      compute eci unit thrust vector

    do i = 1, 3
      sum = 0.0d0

      do j = 1, 3
        sum = sum + qmat(i, j) * utmee(j)
      enddo

      uteci(i) = sum
    enddo

c      define velocity vector match point functions

    ptf(4) = vsc(1) - vmult * (vp(1) + vinfinitiy * uteci(1))
    ptf(5) = vsc(2) - vmult * (vp(2) + vinfinitiy * uteci(2))
    ptf(6) = vsc(3) - vmult * (vp(3) + vinfinitiy * uteci(3))
  else
    ptf(4) = vsc(1) - vmult * vp(1)
    ptf(5) = vsc(2) - vmult * vp(2)
    ptf(6) = vsc(3) - vmult * vp(3)
  end if

end if

if (iphase .eq. 1 .and. iphend .eq. +1) then
c      *****
c      "position match" at end of phase 1
c      *****

    xjdate = xjdateil + time

    call p2000(ip2, xjdate, rp, vp)

c      define position vector match point functions

    ptf(1) = rsc(1) - rp(1) / aunit
    ptf(2) = rsc(2) - rp(2) / aunit
    ptf(3) = rsc(3) - rp(3) / aunit

    if (itraj .eq. 2) then
c      -----
c      rendezvous trajectory
c      -----
c      define velocity vector match point functions

        ptf(4) = vsc(1) - vmult * vp(1)
        ptf(5) = vsc(2) - vmult * vp(2)
        ptf(6) = vsc(3) - vmult * vp(3)
      end if
    end if

return
end

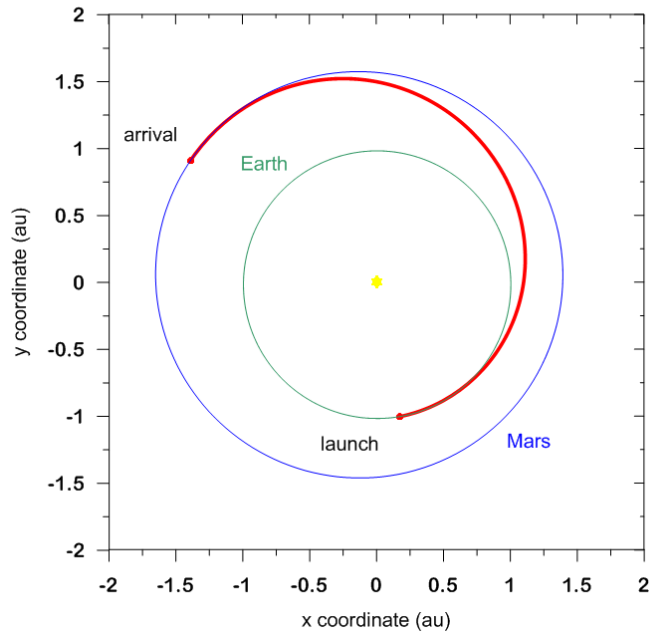
```

# APPENDIX D

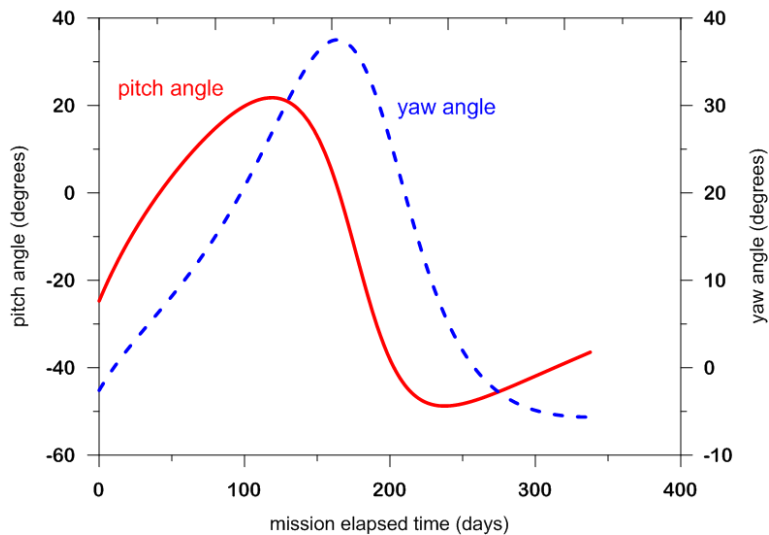
## Example Minimum Transfer Time Trajectory Analysis

This appendix contains graphics and a simulation summary for a typical Earth-to-Mars *minimum transfer time* interplanetary rendezvous mission using chemical propulsion.

Low-thrust Interplanetary Trajectory Analysis  
Earth-to-Mars Minimum Transfer Time  
Mass = 1171.1 kg Thrust = 168.31 N Isp = 3070 s

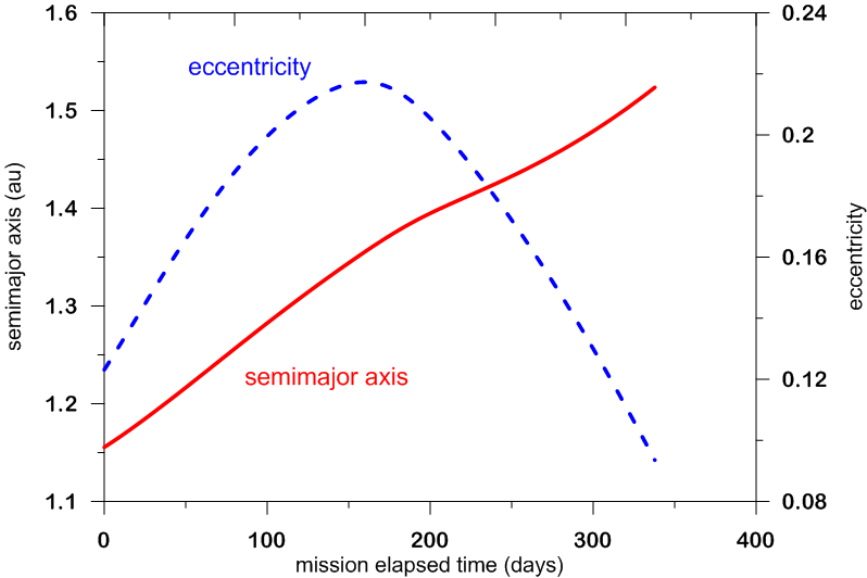


Low-thrust Interplanetary Trajectory Analysis  
Earth-to-Mars Minimum Transfer Time  
Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
(optimal control angles vs. mission elapsed time)

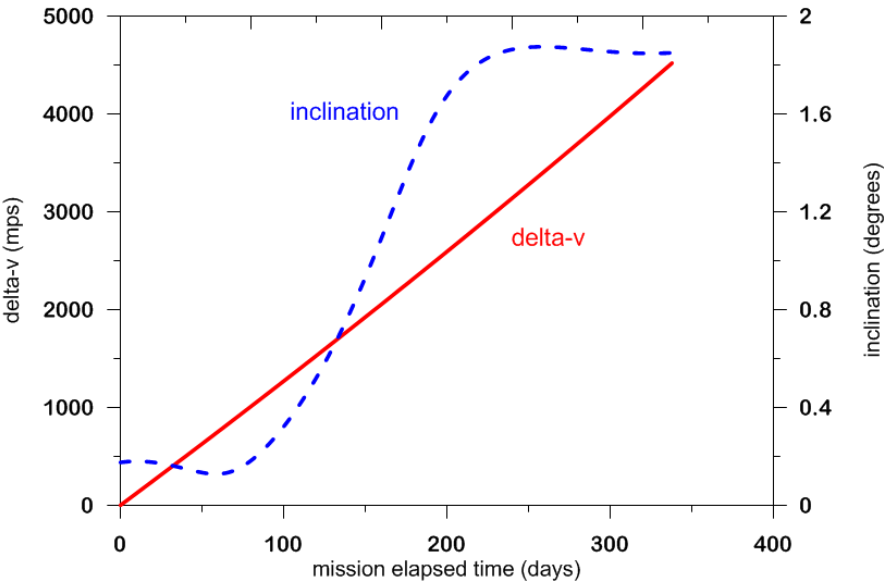


The following two plots illustrate the behavior of the heliocentric semimajor axis, eccentricity, orbital inclination and accumulated delta-v during the interplanetary transfer.

Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Minimum Transfer Time  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
 (semimajor axis and eccentricity vs. mission elapsed time)



Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Mars Minimum Transfer Time  
 Mass = 1171.1 kg Thrust = 168.31 mN Isp = 3070 s  
 (inclination and delta-v vs. mission elapsed time)



Here's the program numerical output for this example using trapezoidal collocation.

```
program ilt_ocs
input file ==> e2m1_min_time.in
rendezvous trajectory
minimum transfer time

LAUNCH CONDITIONS
-----
calendar date      07/01/2005
TDB time           11:26:09.314
TDB julian date    2453552.97649669

launch hyperbola characteristics
(Earth mean equator and equinox of J2000)
-----
right ascension    33.0041733115229      degrees
declination        10.6195596448380      degrees
launch energy      4.625000000000000      (km/sec)**2

heliocentric orbital elements of the departure planet at launch
(Earth mean ecliptic and equinox of J2000)
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1000432031D+01      0.1631213315D-01      0.2205231122D-02      0.3477645136D+03

      raan (deg)      true anomaly (deg)      arglat (deg)      period (days)
0.1163335586D+03      0.1755526005D+03      0.1633171141D+03      0.3654936272D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.2549751741D+08      -.1499437805D+09      0.1680522702D+04      0.1520962219D+09

      vx (kps)      vy (kps)      vz (kps)      vmag (kps)
0.2888957025D+02      0.4874377540D+01      -.1079752277D-02      0.2929789799D+02

heliocentric orbital elements of the spacecraft at launch
(Earth mean ecliptic and equinox of J2000)
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1155321712D+01      0.1230681108D+00      0.1749818469D+00      0.1943199109D+03

      raan (deg)      true anomaly (deg)      arglat (deg)      period (days)
0.9985796247D+02      0.3454727984D+03      0.1797927093D+03      0.4535788853D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.2549751741D+08      -.1499437805D+09      0.1680522702D+04      0.1520962219D+09
```

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.3066222342D+02	0.6088377602D+01	-.9544380167D-01	0.3126098841D+02

ARRIVAL CONDITIONS

```

-----
calendar date      06/04/2006
TDB time          07:31:46.918
TDB julian date   2453890.81373747

flight time       337.837240782112    days
spacecraft mass   1007.91804464375    kilograms
delta-v          4517.66684906135    meters/second

```

heliocentric conditions of the destination celestial body at arrival  
(Earth mean ecliptic and equinox of J2000)

```

-----
sma (au)          eccentricity      inclination (deg)   argper (deg)
0.1523593257D+01  0.9348701651D-01  0.1849328659D+01  0.2865824155D+03

raan (deg)        true anomaly (deg)  arglat (deg)       period (days)
0.4953748812D+02  0.1705051279D+03  0.9708754335D+02  0.6869134021D+03

rx (km)           ry (km)            rz (km)            rmag (km)
-.2077412613D+09  0.1368309357D+09  0.7970392623D+07  0.2488828314D+09

vx (kps)          vy (kps)           vz (kps)           vmag (kps)
-.1241641781D+02  -.1816677609D+02  -.7563626703D-01  0.2200465645D+02

```

heliocentric conditions of the spacecraft at arrival  
(Earth mean ecliptic and equinox of J2000)

```

-----
sma (au)          eccentricity      inclination (deg)   argper (deg)
0.1523593257D+01  0.9348701651D-01  0.1849328659D+01  0.2865824155D+03

raan (deg)        true anomaly (deg)  arglat (deg)       period (days)
0.4953748812D+02  0.1705051279D+03  0.9708754335D+02  0.6869134021D+03

rx (km)           ry (km)            rz (km)            rmag (km)
-.2077412613D+09  0.1368309357D+09  0.7970392623D+07  0.2488828314D+09

vx (kps)          vy (kps)           vz (kps)           vmag (kps)
-.1241641781D+02  -.1816677609D+02  -.7563626703D-01  0.2200465645D+02

```

INTEGRATED SOLUTION WITH OPTIMAL CONTROL

```

=====
tzero             -8.52350330775228    days
tfinal           329.313737474360    days

```

final heliocentric position vector and magnitude errors

delta rx	0.982853084802628	kilometers
delta ry	-1.99446716904640	kilometers
delta rz	-0.300119932740927	kilometers
delta rmag	2.24365136523891	kilometers

final heliocentric velocity vector and magnitude errors

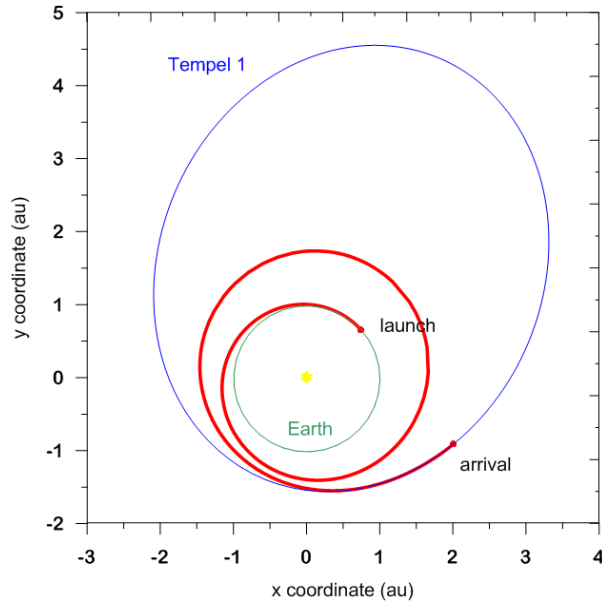
delta vx	1.845999761940220E-007	kilometers/second
delta vy	-2.231874027813774E-007	kilometers/second
delta vz	-2.566047183072406E-008	kilometers/second
delta vmag	2.907717795552135E-007	kilometers/second
delta-v	4517.66684916448	meters/second

# APPENDIX F

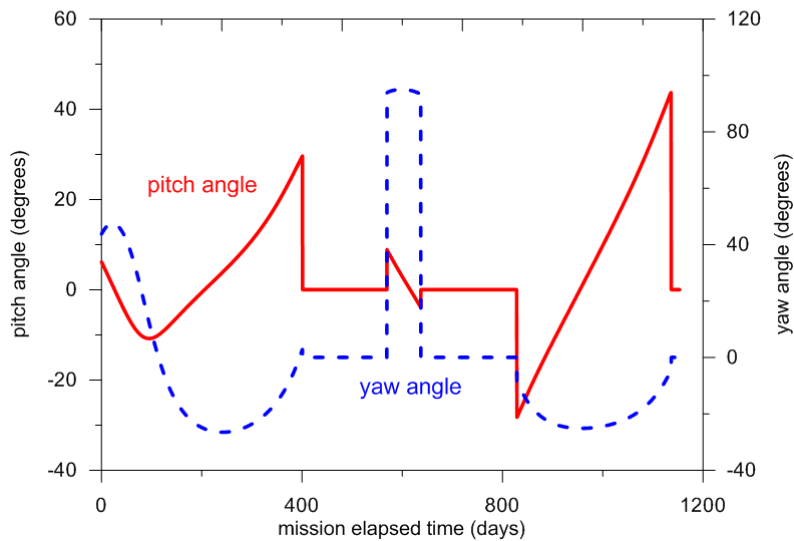
## Example Comet Rendezvous Trajectory Analysis

This appendix contains graphics and a simulation summary for a maximum final mass, interplanetary rendezvous mission from Earth to the comet Tempel 1 using solar electric propulsion.

Low-thrust Interplanetary Trajectory Analysis  
Earth-to-Tempel 1 Maximum Final Mass  
Mass = 576 kg SEP Power = 10 kw @ 1 AU

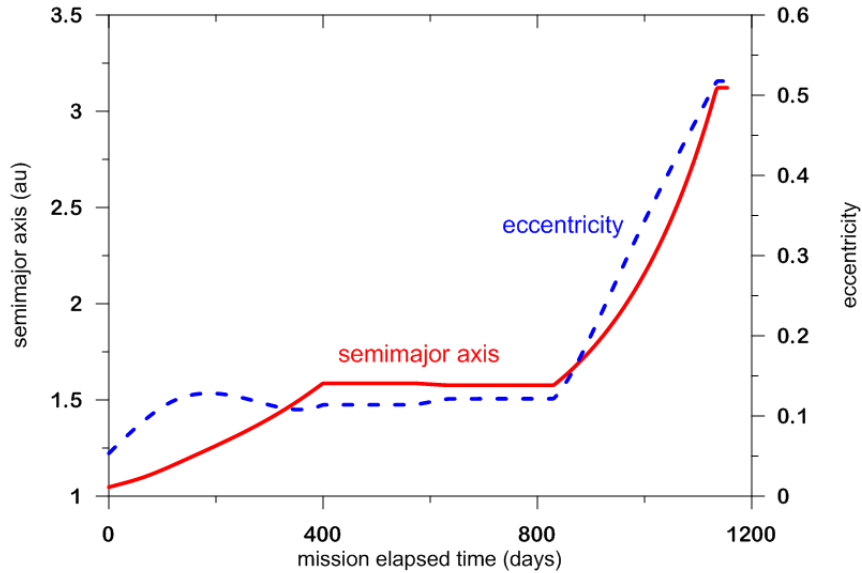


Low-thrust Interplanetary Trajectory Analysis  
Earth-to-Tempel 1 Maximum Final Mass  
Mass = 576 kg SEP Power = 10 kw @ 1 AU  
(optimal control angles vs. mission elapsed time)

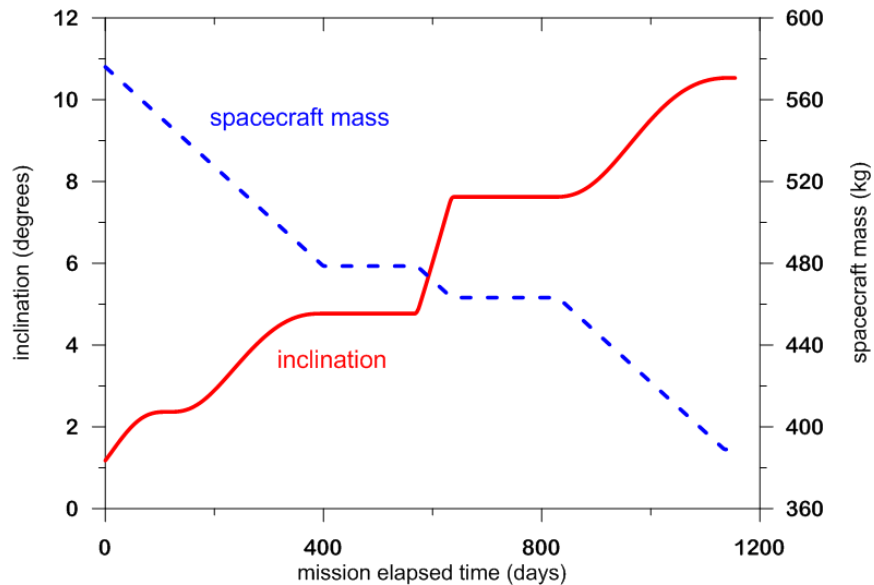


The following two plots illustrate the behavior of the semimajor axis, eccentricity, inclination and spacecraft mass during the interplanetary transfer.

Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Tempel 1 Maximum Final Mass  
 Mass = 576 kg SEP Power = 10 kw @ 1 AU  
 (semimajor axis and inclination vs. mission elapsed time)

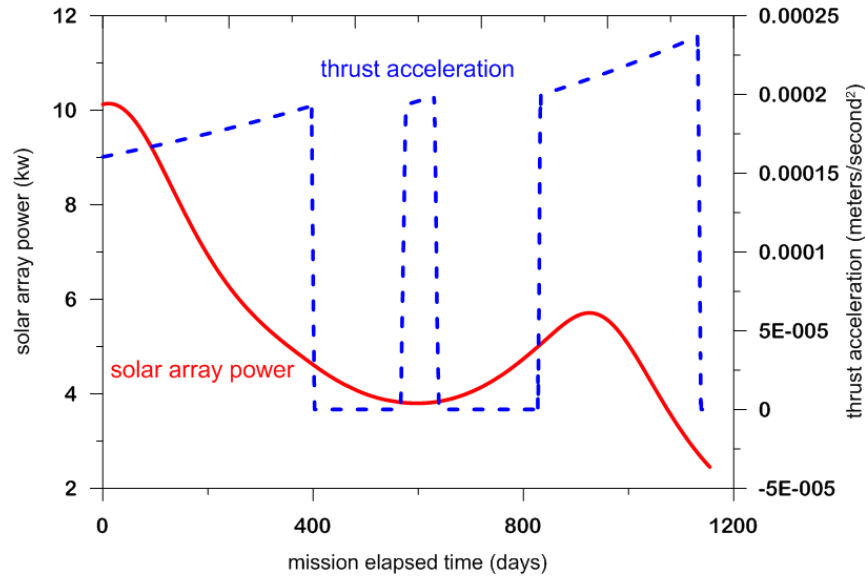


Low-thrust Interplanetary Trajectory Analysis  
 Earth-to-Tempel 1 Maximum Final Mass  
 Mass = 576 kg SEP Power = 10 kw @ 1 AU  
 (inclination and mass vs. mission elapsed time)



The final plot illustrates the behavior of the available solar array power and the thrust acceleration during the interplanetary transfer.

**Low-thrust Interplanetary Trajectory Analysis**  
**Earth-to-Tempel 1 Maximum Final Mass**  
 Mass = 576 kg SEP Power = 10 kw @ 1 AU  
 (power and thrust acceleration vs. mission elapsed time)



Here's the program numerical output for this example using trapezoidal collocation.

```

program ilt_ocs
input file ==> tempell_max_payload.in
rendezvous trajectory
maximum payload

LAUNCH CONDITIONS
-----

calendar date      11/04/2002
TDB time           01:07:29.851
TDB julian date    2452582.54687327

launch hyperbola characteristics
(Earth mean equator and equinox of J2000)
-----

right ascension    145.002380710242    degrees
declination        61.9245330434051    degrees
launch energy      0.8000000000000000    (km/sec)**2
  
```

heliocentric orbital elements of the departure planet at launch  
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1000926361D+01  0.1706849123D-01  0.1505127320D-02  0.4459557462D+02

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.5564889164D+02  0.3011632584D+03    0.3457588330D+03    0.3657645552D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.1112899240D+09  0.9814193557D+08  -.9589024794D+03  0.1483822318D+09

      vx (kps)     vy (kps)     vz (kps)     vmag (kps)
-.2019384909D+02  0.2224169878D+02  0.7676480465D-03  0.3004138323D+02
```

heliocentric orbital elements of the spacecraft at launch  
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1046435806D+01  0.5329831146D-01  0.1174576604D+01  0.1259590030D+02

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.4142578374D+02  0.3473860368D+03    0.3599819371D+03    0.3909914983D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.1112899240D+09  0.9814193557D+08  -.9589024815D+03  0.1483822318D+09

      vx (kps)     vy (kps)     vz (kps)     vmag (kps)
-.2053867962D+02  0.2277712479D+02  0.6287890598D+00  0.3067621473D+02
```

ARRIVAL CONDITIONS

```
-----
calendar date      12/31/2005
TDB time           09:27:25.212
TDB julian date    2453735.89404180

flight time        1153.34716852868      days
spacecraft mass    388.957318524463      kilograms
delta-v            12848.8672295688     meters/second
```

heliocentric conditions of the destination celestial body at arrival  
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.3121939735D+01  0.5175674802D+00  0.1052963297D+02  0.1788381138D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.6894110071D+02  0.8761888543D+02  0.2664569992D+03  0.2014815113D+04

      rx (km)      ry (km)      rz (km)      rmag (km)
0.2990939734D+09  -.1373302895D+09  -.6105302696D+08  0.3347301681D+09
```

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.1746174759D+02	0.1412553576D+02	-.2085471820D+01	0.2255643106D+02

heliocentric conditions of the spacecraft at arrival  
(Earth mean ecliptic and equinox of J2000)

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.3121939735D+01	0.5175674802D+00	0.1052963297D+02	0.1788381138D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.6894110071D+02	0.8761888543D+02	0.2664569992D+03	0.2014815113D+04

rx (km)	ry (km)	rz (km)	rmag (km)
0.2990939734D+09	-.1373302895D+09	-.6105302696D+08	0.3347301681D+09

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.1746174759D+02	0.1412553576D+02	-.2085471820D+01	0.2255643106D+02

INTEGRATED SOLUTION WITH OPTIMAL CONTROL  
=====

tzero	-26.9531267251387	days
tfinal	1126.39404180354	days

final heliocentric position vector and magnitude errors

delta rx	4.34277135133743	kilometers
delta ry	-27.3916816115379	kilometers
delta rz	8.84477781504393	kilometers
delta rmag	29.1100322760961	kilometers

final heliocentric velocity vector and magnitude errors

delta vx	2.650010788585178E-006	kilometers/second
delta vy	-1.123592202390000E-006	kilometers/second
delta vz	-3.619323889481052E-007	kilometers/second
delta vmag	2.901036309848453E-006	kilometers/second
delta-v	12879.4667192551	meters/second

Here's the input data file for this example.

```
*****
** interplanetary trajectory optimization
** patched-conic, n-body heliocentric motion
** low-thrust maneuvers
** Earth-to-Tempel 1 - March 21, 2011
*****
```

type of optimization (1 = maximum payload, 2 = minimum transfer time)  
1

trajectory type (1 = flyby, 2 = rendezvous)  
2

```

propulsion type (1 = chemical, 2 = SEP)
2

launch energy ([km/sec]**2)
0.8

initial spacecraft mass (kilograms)
576.0

thrust magnitude (newtons)
0.1

specific impulse (seconds)
3370.0

solar array power at 1 AU (kw)
10.0

solar array minimum and maximum power (kw)
0.649, 2.6

solar array power coefficients
1.1063, 0.1495, -0.299, -0.0432, 0.0

SEP thrust magnitude coefficients
-1.9137, 36.242, 0.0, 0.0, 0.0

SEP propellant flow rate coefficients
0.47556, 0.90209, 0.0, 0.0, 0.0

lower bound for throttle setting (0 <= bound <= 1)
0.0

upper bound for throttle setting (0 <= bound <= 1)
1.0

*****
* LAUNCH CONDITIONS *
*****

launch calendar date initial guess (month, day, year)
12,1,2002

launch date search boundary (days)
-30, +30

*****
* launch planet *
*****
1 = Mercury
2 = Venus
3 = Earth
4 = Mars
5 = Jupiter
6 = Saturn
7 = Uranus
8 = Neptune
9 = Pluto
-----
3

*****
* ARRIVAL CONDITIONS *
*****

arrival calendar date initial guess (month, day, year)

```

12,3,2005

arrival date search boundary (days)  
-60, +60

\*\*\*\*\*  
\* arrival celestial body \*  
\*\*\*\*\*

- 1 = Mercury
- 2 = Venus
- 3 = Earth
- 4 = Mars
- 5 = Jupiter
- 6 = Saturn
- 7 = Uranus
- 8 = Neptune
- 9 = Pluto
- 0 = asteroid/comet

-----  
0

\*\*\*\*\*  
\* asteroid/comet orbital elements \*  
\* (heliocentric, ecliptic J2000) \*  
\*\*\*\*\*

asteroid/comet name  
Tempel 1

calendar date of perihelion passage (month, day, year)  
7,5.31630136,2005

perihelion distance (au)  
1.50612525322912

orbital eccentricity (nd)  
0.517567480182153

orbital inclination (degrees)  
10.5296329695782

argument of perihelion (degrees)  
178.83811381255987

longitude of the ascending node (degrees)  
68.94110070770958

\*\*\*\*\*  
\* initial guess/restart option \*  
\*\*\*\*\*

- 1 = numerical integration
- 2 = binary data file

-----  
1

name of initial guess/restart input data file  
tempell\_max\_payload.rsbin

\*\*\*\*\*  
\* restart data file option \*  
\*\*\*\*\*

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

```

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

number of user-defined nodes or print step size in solution data file
1

name of solution output file
tempell_max_payload.csv

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

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

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

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

maximum number of mesh refinement iterations
20

maximum number of function evaluations
50000

maximum number of algorithm iterations
5000

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

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

*****
differential equation output
-----

```

1 = none  
2 = terse  
3 = standard  
4 = interpretive  
5 = diagnostic

-----

1

\*\*\*\*\*

user-defined output

-----

input no to ignore

-----

a0b0c0d0e0f0g0h0i0j2k0l0m0n0o0p0q0r0

\*\*\*\*\*

\* optimal control configuration options

\*\*\*\*\*

read an optimal control configuration file (yes or no)

no

name of optimal control configuration file

tempell\_config.txt

create an optimal control configuration file (yes or no)

no

name of optimal control configuration file

tempell\_config.txt

# APPENDIX G

## Typical Configuration File

The `ilt_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 **INSOCX** routine in section 6.2, *Subprograms for Optimal Control*, and the **INSNLP** routine in Section 2.2, *Subprograms for Optimization* of the *AMA\_OC* software user's manual. Please note that the `ilt_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 `ilt_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=0
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
NPATAUX=100
```

NSSWCH=-1  
SOCOUT=A0B0C0D0E0F0G0H0I0J2K0L0M0N0O0P0Q0R0S1T0U0V0W0X0Y0Z0  
SPRTHS=SPARSE  
NLPALG=SNLPMN  
NLPOMR=M  
KEYDPL=.lueiLUE  
RHSTMP=RHSTMPLT  
RSTFIL=socx.restart  
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=500000  
INSNLP:MNSAME=2  
INSNLP:NEWTON=0  
INSNLP:NITMAX=50000  
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