

Program ipto_ocs

Ballistic Interplanetary Trajectory Optimization

This document is the user's manual for a Fortran computer program called `ipto_ocs` that uses optimal control software distributed by Applied Mathematical Analysis to solve the classic one impulse flyby and two-impulse rendezvous ballistic interplanetary trajectory optimization problems. The software attempts to minimize the launch delta-v, the arrival delta-v or the total delta-v for the interplanetary transfer. The type of trajectory optimization is specified by the user.

The important features of this scientific simulation are as follows:

- one impulse, *patched-conic* interplanetary *flyby* trajectory modeling
- two impulse, *patched-conic* interplanetary *rendezvous* trajectory modeling
- n-body heliocentric, inertial cartesian equations of motion
- elliptical, non-coplanar asteroid and comet orbits
- 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 (www.siam.org).

The `ipto_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

`AMA_OC` will use this information to *automatically* transcribe the user's problem and perform the optimization using a sparse nonlinear programming (NLP) method. The `ipto_ocs` software allows the user to select the type of 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:

```
ipto_ocs mars03.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 ipt_ocs           *
*                                     *
*   ballistic interplanetary         *
*   trajectory optimization           *
*                                     *
*           May 6, 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 `ipto_ocs` computer program is “data-driven” by a simple user-created text file. The following is a typical input or “simulation definition” file used by the software. This example is an Earth-to-Mars ballistic rendezvous trajectory that minimizes the total delta-v for the mission. Please note for a flyby trajectory input file (trajectory type = 1), the only valid optimization option is 1 = minimize launch delta-v.

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).

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.

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.

```
*****
** impulsive delta-v interplanetary trajectory optimization
** patched-conic heliocentric motion - ipt_ocs
** Mars '03 - mars03.in
** May 6, 2011
*****
```

The first program input is an integer that defines the type of delta-v optimization. Please note that option 4, no optimization simply solves the n-body, orbital two-point boundary value problem.

```
*****
* simulation type *
*****
1 = minimize launch delta-v
2 = minimize arrival delta-v
3 = minimize total delta-v
4 = no optimization
-----
3
```

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

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

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 calendar date initial guess (month, day, year)
6,1,2003
```

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 Δt_l and Δt_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 Δt_l and Δt_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 should be set to 0.

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)
-30, +30
```

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 search interval and the arrival planet/comet/asteroid.

```
arrival calendar date initial guess (month, day, year)
12,1,2003
```

```
arrival date search boundary (days)
-30, +30
```

```
*****
* 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. The calendar date of perihelion passage should be with respect to the TDB time system.

```
*****
* asteroid/comet classical orbital elements *
* (heliocentric, Earth mean 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 (non-dimensional)
0.517491
```

```
orbital inclination (degrees)
10.5301
```

```
argument of perihelion (degrees)
178.8390
```

```
longitude of the ascending node (degrees)
68.9734
```

This next integer 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 = 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 (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 B of this document for a description of the information written to this file.

```

name of solution output file
mars03.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.

```

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 AMA 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 AMA 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 AMA 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
```

Optimal Control Solution

The following is the program output created by the `ipto_ocs` simulation for this example. The first part of the solution display summarizes important information about the type of trajectory and the type of optimization and the launch conditions. Please note that the launch hyperbola coordinates are with respect to the Earth mean equator and equinox of J2000 (EME2000) system. The second part of the display summarizes the arrival conditions and characteristics of the planetary and heliocentric transfer orbits. Please see Appendix A of this document for additional details about this information.

program ipto_ocs
=====

rendezvous trajectory

minimize total delta-v

DE421 ephemeris

DEPARTURE CONDITIONS
=====

calendar date 06/06/2003
TDB time 08:18:17.36
TDB julian date 2452796.84603420

heliocentric delta-v vector and magnitude
(Earth mean ecliptic and equinox of J2000)

launch delta-vx 2900.35549073551 meters/second
launch delta-vy -616.714043478893 meters/second
launch delta-vz -40.1696083298423 meters/second

launch delta-v 2965.46990905271 meters/second

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

right ascension 349.265305839434 degrees
declination -5.46005203692079 degrees
C3 8.79401178149709 (km/sec)**2

heliocentric orbital elements and state vector of the departure planet
(Earth mean ecliptic and equinox of J2000)

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1000231776D+01	0.1633787580D-01	0.3276160283D-03	0.2719030375D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.1902549286D+03	0.1530415819D+03	0.6494461938D+02	0.3653838922D+03
rx (km)	ry (km)	rz (km)	rmag (km)
-.3877865233D+08	-.1467666196D+09	0.7863271593D+03	0.1518032427D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.2832127123D+02	-.7711221751D+01	0.7221819658D-04	0.2935229710D+02

heliocentric orbital elements and state vector of the spacecraft
(Earth mean ecliptic and equinox of J2000)

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1259528064D+01	0.1943611232D+00	0.7109964954D-01	0.1789331987D+03

raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.7543871484D+02	0.8276342515D+00	0.1797608329D+03	0.5163095994D+03
rx (km)	ry (km)	rz (km)	rmag (km)
-.3877865235D+08	-.1467666197D+09	0.7863271176D+03	0.1518032428D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.3122162673D+02	-.8327935797D+01	-.4009739015D-01	0.3231324958D+02

ARRIVAL CONDITIONS
=====

calendar date 12/27/2003
TDB time 16:57:09.05
TDB julian date 2453001.20635469

heliocentric delta-v vector and magnitude
(Earth mean ecliptic and equinox of J2000)

arrival delta-vx	2021.79887865832	meters/second
arrival delta-vy	-1614.33782179448	meters/second
arrival delta-vz	-779.406642586644	meters/second
arrival delta-v	2702.07920371116	meters/second
C3	7.30123202312836	(km/sec)**2

heliocentric orbital elements and state vector of the planet/asteroid/comet
(Earth mean ecliptic and equinox of J2000)

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1523678538D+01	0.9354080419D-01	0.1849372004D+01	0.2865170844D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.4954092332D+02	0.7248490912D+02	0.3590019936D+03	0.6869710761D+03
rx (km)	ry (km)	rz (km)	rmag (km)
0.1454912476D+09	0.1646993683D+09	-.1235263971D+06	0.2197580494D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.1723361516D+02	0.1810898471D+02	0.8028150130D+00	0.2501154394D+02

heliocentric orbital elements and state vector of the spacecraft
(Earth mean ecliptic and equinox of J2000)

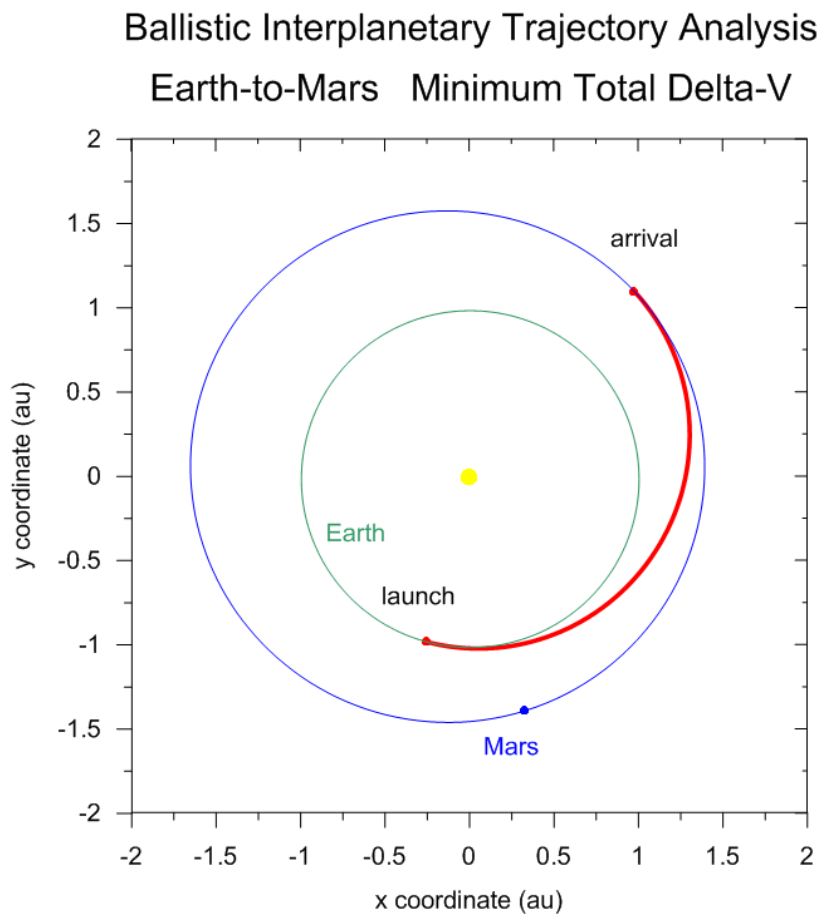
sma (au)	eccentricity	inclination (deg)	argper (deg)
0.1523678539D+01	0.9354080441D-01	0.1849372004D+01	0.2865170849D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (days)
0.4954092332D+02	0.7248490871D+02	0.3590019936D+03	0.6869710771D+03
rx (km)	ry (km)	rz (km)	rmag (km)
0.1454912476D+09	0.1646993683D+09	-.1235263972D+06	0.2197580495D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.1723361516D+02	0.1810898471D+02	0.8028150132D+00	0.2501154395D+02

The simulation also provides the total delta-v for the mission along with the transfer time in days.

total delta-v	5667.54911276387	meters/second
transfer time	204.360320485197	days

The `ipto_ocs` software will create two comma-separated-variable (csv) output files. The first file contains the heliocentric, ecliptic state vector of the spacecraft and the second file (`planets.csv`) contains the state vectors of the planet and the destination celestial body. These data files can be used to create graphic displays of the trajectory. Appendix A provides additional information about the contents of these two data files.

The following is the transfer trajectory for this example. It is a view of the trajectory and planetary orbits from the north pole of the ecliptic looking down on the ecliptic plane. The transfer trajectory is red and the scales are in astronomical units.



Verification of the Optimal Control Solution

The optimal control solution determined by the `AMA_OC` software can be verified by numerically integrating the n-body orbital equations of motion with the OC-computed initial departure conditions and the optimal delta-v vectors. This is equivalent to solving an initial value problem (IVP) that uses the `AMA_OC` optimal solution. This part of the `ipto_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. This display includes the final heliocentric orbital elements of the spacecraft along with the position and velocity “matching” errors.

```

=====
verification of optimal control solution
=====

heliocentric orbital elements and state vector of the spacecraft at arrival
(Earth mean ecliptic and equinox of J2000)
-----

      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1523678462D+01    0.9354076752D-01    0.1849372014D+01    0.2865170728D+03

      raan (deg)    true anomaly (deg)    arglat (deg)      period (days)
0.4954092238D+02    0.7248492066D+02    0.3590019934D+03    0.6869710250D+03

      rx (km)      ry (km)      rz (km)      rmag (km)
0.1454912486D+09    0.1646993631D+09    -.1235264148D+06    0.2197580462D+09

      vx (kps)      vy (kps)      vz (kps)      vmag (kps)
-.1723361512D+02    0.1810898443D+02    0.8028150120D+00    0.2501154372D+02

final heliocentric position vector and magnitude errors

delta rx      1.04274883866310      kilometers
delta ry      -5.17958366870880      kilometers
delta rz      -1.770855819631834E-002 kilometers

delta rmag      5.28353344975894      kilometers

final heliocentric velocity vector and magnitude errors

delta vx      3.511720336746293E-008 kilometers/second
delta vy      -2.801622756010147E-007 kilometers/second
delta vz      -9.508036580285761E-010 kilometers/second

delta vmag      2.823561982140410E-007 kilometers/second

```

Creating an Initial Guess

An initial guess for the launch and arrival impulsive delta- v vectors can be determined from the solution of the Lambert two-point boundary-value problem (TPBVP). Lambert’s Theorem states that the time to traverse a trajectory depends only upon the length of the semimajor axis a of the transfer trajectory, the sum $r_i + r_f$ of the distances of the initial and final positions relative to a central body, and the length c of the chord joining these two positions.

The Lambert solution that initializes the `ipto_ocs` software uses the user’s initial guess for launch and arrival dates. 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 `AMA_OC` to use the Dormand-Prince variable step size numerical method to solve the initial value problem.

The algorithm used in this computer program to solve the two-body Lambert problem is based on the method described in “A Procedure for the Solution of Lambert’s Orbital Boundary-Value Problem” by R. H. Gooding, *Celestial Mechanics and Dynamical Astronomy* **48**: 145-165, 1990. This iterative solution is valid for elliptic, parabolic and hyperbolic transfer orbits which may be either posigrade or retrograde, and involve one or more revolutions about the central body.

Problem setup

This section provides additional details about the software implementation. For good scaling during the optimization, the time unit used in all internal calculations is days, position is expressed in astronomical units, and the unit for velocity and delta-v 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 bounds on the actual dates found during the optimization process. For any guess for launch time t_L and user-defined launch time search bound Δt_L , the launch time t is constrained as follows:

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

Likewise, for any guess for arrival time t_A and user-defined arrival time bound Δt_A , the arrival time t is constrained as follows:

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

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

(2) Performance index – minimize delta-v

The objective function or performance index J for this simulation is one of three possible delta-v’s. For this classic trajectory optimization problem, this index is simply

$$J = \Delta V$$

where ΔV is either the launch delta-v, arrival delta-v or the total delta-v. The value of the `maxmin` indicator in the `AMA_OC` software tells the program whether the user is minimizing or maximizing the performance index. The type of delta-v is selected by the user.

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

For any launch time t_L , the optimal solution for a rendezvous trajectory must satisfy the following state vector boundary conditions (equality constraints) at launch:

$$\mathbf{r}_{s/c}(t_L) - \mathbf{r}_p(t_L) = 0$$

$$\mathbf{v}_{s/c}(t_L) - \{\mathbf{v}_p(t_L) + \Delta\mathbf{v}(t_L)\} = 0$$

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 *impulsive* heliocentric delta-v vector required at launch.

(4) Point functions – position and/or velocity vector “matching” at arrival

For any arrival time t_A , the optimal solution for a rendezvous trajectory must satisfy the following state vector boundary conditions at arrival:

$$\mathbf{r}_{s/c}(t_A) - \mathbf{r}_p(t_A) = 0$$

$$\mathbf{v}_{s/c}(t_A) - \{\mathbf{v}_p(t_A) + \Delta\mathbf{v}(t_A)\} = 0$$

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, and $\Delta\mathbf{v}$ is the *impulsive* heliocentric delta-v vector required at arrival.

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.

The components of the departure and arrival impulsive delta-v's are the optimization parameters used by the AMA_OC software suite to solve this trajectory problem.

Technical Discussion

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 , μ_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 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 third-body acceleration can now be expressed as

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

The first-order system of equations required by the *AMA_OC* software can be created from the second-order system by the method of *order reduction*. With the following definitions,

$$\begin{aligned} y_1 &= r_x & y_2 &= r_y & y_3 &= r_z \\ y_4 &= v_x & y_5 &= v_y & y_6 &= v_z \end{aligned}$$

where v_x, v_y, v_z are the velocity vector components of the spacecraft, the first-order system of differential equations is given by

$$\begin{aligned} \dot{y}_1 &= v_x & \dot{y}_2 &= v_y & \dot{y}_3 &= v_z \\ \dot{y}_4 &= -\mu_s \frac{r_x}{r^3} + a_x & \dot{y}_5 &= -\mu_s \frac{r_y}{r^3} + a_y & \dot{y}_6 &= -\mu_s \frac{r_z}{r^3} + a_z \end{aligned}$$

In these equations, μ_s is the gravitational constant of the sun, and a_x, a_y and a_z are the x, y and z gravitational contributions of the planets. All point-mass planetary perturbations except those due to the launch and arrival planets are included in the equations of motion. The equations described here are coded in the right-hand-side subroutine required by the *AMA_OC* software.

The software models the planetary coordinates using the DE421 model from JPL.

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.

In order to model ballistic interplanetary missions involving asteroids and comets, the classical 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 μ_s is the gravitational constant of the sun, a is the semimajor axis of the celestial body's heliocentric orbit, 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. 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 equation we want to solve is

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

which has the first three 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 ε 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

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

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 three 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 Δ, Δ^* , and so forth is the same. The true anomaly for hyperbolic orbits is determined with this next set of equations:

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

The true anomaly is then determined from a four quadrant inverse tangent evaluation of these two equations.

The ΔV 's required at launch and arrival are simply the differences between the velocity on the optimal transfer trajectory and the heliocentric velocities of the two celestial objects. If we treat each planet as a point mass and assume *impulsive* maneuvers, the *planet-centered* magnitude and direction of the required maneuvers are given by the two vector equations:

$$\Delta \mathbf{V}_L = \mathbf{V}_{T_L} - \mathbf{V}_{P_L}$$

$$\Delta \mathbf{V}_A = \mathbf{V}_{P_A} - \mathbf{V}_{T_A}$$

where

\mathbf{V}_{T_L} = heliocentric velocity vector of the transfer trajectory at launch

\mathbf{V}_{T_A} = heliocentric velocity vector of the transfer trajectory at arrival

\mathbf{V}_{P_L} = heliocentric velocity vector of the launch planet

\mathbf{V}_{P_A} = heliocentric velocity vector of the arrival planet

The scalar magnitude of each maneuver is also called the “hyperbolic excess velocity” or V_∞ at launch and arrival. The hyperbolic excess velocity is the speed of the spacecraft relative to each planet or celestial body at an *infinite* distance from the planet. Furthermore, the *energy* or C_3 at launch or arrival is equal to V_∞^2 for the respective maneuver. C_3 is also equal to twice the orbital energy per unit mass (the specific orbital energy).

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 Cartesian components of the V_∞ velocity vector.

The right ascension of the asymptote is determined from

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

and the geocentric declination of the asymptote is given by

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

where $\Delta \hat{V}_x$, $\Delta \hat{V}_y$ and $\Delta \hat{V}_z$ are the x, y and z components of the unit ΔV vector. The right ascension is computed using a four quadrant inverse tangent function.

In this computer program the heliocentric planetary coordinates and therefore the ΔV vectors are computed in the Earth mean ecliptic and equinox of J2000 coordinate system. In order to determine the orientation of the departure and arrival hyperbolas, these ΔV vectors must be transformed to the equatorial frame.

The required transformation is given by

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

where $\Delta \mathbf{V}_{ec}$ is the delta-velocity vector in the ecliptic frame, and $\Delta \mathbf{V}_{eq}$ is the delta-velocity vector in the equatorial frame.

The transformation of vectors from the equatorial to the ecliptic system involves the transpose of this matrix according to

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

References and Bibliography

“Modern Astrodynamics”, Victor R. Bond and Mark C. Allman, Princeton University Press, 1996.

“Interplanetary Mission Design Handbook, Volume 1, Part 2”, JPL Publication 82-43, September 15, 1983.

“Rapid Solar Sail Rendezvous Missions to Asteroid 99942 Apophis”, Giovanni Mengali and Alessandro A. Quarta, *AIAA Journal of Spacecraft and Rockets*, Vol. 46, No. 1, January-February 2009, pp. 134-140.

“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.

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 `ipto_ocs` software. All output except the coordinates of the launch hyperbola 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
right ascension = right ascension of the launch hyperbola in degrees
declination = declination of the launch hyperbola in degrees
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-vx = x-component of the impulsive velocity vector in kilometers/second
delta-vy = y-component of the impulsive velocity vector in kilometers/second
delta-vz = z-component of the impulsive velocity vector in kilometers/second
delta-v = scalar magnitude of the impulsive delta-v in kilometers/seconds
energy (km/sec) = twice specific orbital energy in $\text{km}(\text{sec})^{**2}$

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
rx (au) = x-component of spacecraft position vector in astronomical units
ry (au) = y-component of spacecraft position vector in astronomical units
rz (au) = z-component of spacecraft position vector in astronomical units
rmag (au) = heliocentric radius magnitude of spacecraft in astronomical units

vx (km/sec) = x-component of spacecraft velocity vector in kilometers per second
vy (km/sec) = y-component of spacecraft velocity vector in kilometers per second
vz (km/sec) = z-component of spacecraft velocity vector in kilometers per second
vmag (km/sec) = heliocentric velocity of spacecraft in kilometers per second
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
period (days) = orbital period in days

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 position vector in astronomical units
rp1-y (au) = y-component of the launch planet position vector in astronomical units
rp1-z (au) = z-component of the launch planet position vector in astronomical units
rp2-x (au) = x-component of the destination body position vector in astronomical units
rp2-y (au) = y-component of the destination body position vector in astronomical units
rp2-z (au) = z-component of the destination body 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 `ipto_ocs` computer program.

- ipto_ocs.f** - AMA_OC main executive program
- atan3.for** - four quadrant inverse tangent function
- eci2orb.for** - convert eci position and velocity vectors to classical orbital elements subroutine
- gdate.for** - compute calendar date from Julian date 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
- glambert.for** - solve Lambert's problem using Gooding's method subroutine
- odeinp.for** - AMA_OC simulation input subroutine
- odepf.for** - AMA_OC point functions subroutine
- odeprt.for** - AMA_OC print subroutine - creates comma-separated-variable file
- oderhs.for** - AMA_OC 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
- slp96.for** - read and interpolate SLP96 ephemeris subroutine
- svprint.for** - subroutine that displays position and velocity vectors
- 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 77 routine and illustrates typical programming conventions used in the `ipto_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 functions
c     and scalar delta-v objective function

c     *****

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

      include 'socscom1.inc'

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

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

      dimension rp(3), vp(3)

      if (iphase .eq. 1 .and. iphend .eq. -1) then

c         *****
c         "position & velocity match" at beginning of phase
c         for either rendezvous or flyby program option
c         *****

      xjdate = time + xjdateil

      call p2000(ip1, xjdate, rp, vp)

c     position match

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

c     velocity match

      ptf(4) = ydyn(4) - (vmult * vp(1) + parm(1))
      ptf(5) = ydyn(5) - (vmult * vp(2) + parm(2))
      ptf(6) = ydyn(6) - (vmult * vp(3) + parm(3))

      if (iopt .eq. 1 .or. iopt .eq. 3) then

c         *****
c         launch delta-v contribution to objective function
c         *****
```

```

        ptf(7) = sqrt(parm(1)**2 + parm(2)**2 + parm(3)**2)
    end if
end if

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

    xjdate = time + xjdateil

    call p2000(ip2, xjdate, rp, vp)

c    position match

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

    if (itraj .eq. 2) then

c      -----
c      rendezvous option - include velocity match
c      -----

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

    if (iopt .eq. 2 .or. iopt .eq. 3) then

c      *****
c      arrival delta-v contribution to objective function
c      *****

        ptf(7) = sqrt(parm(4)**2 + parm(5)**2 + parm(6)**2)

    end if

end if

return
end

```

APPENDIX E

Earth-to-Tempel 1 Flyby Analysis

This appendix summarizes typical trajectory characteristics of a ballistic flyby mission from Earth to the comet Tempel 1. This simulation minimizes the magnitude of the launch delta-v at the Earth departure.

Please note for a flyby trajectory input file (trajectory type = 1), the only valid delta-v optimization option is 1 = minimize launch delta-v.

Here's the ipto_ocs input data file for this example.

```
*****
** impulsive delta-v interplanetary trajectory optimization
** ballistic patched-conic heliocentric motion - ipto_ocs
** Earth-to-Tempel 1 flyby - tempel1.in
** May 6, 2011
*****

*****
* simulation type *
*****
 1 = minimize launch delta-v
 2 = minimize arrival delta-v
 3 = minimize total delta-v
 4 = no optimization
-----
1

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

launch calendar date initial guess (month, day, year)
12,1,2004

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

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

arrival calendar date initial guess (month, day, year)
7,1,2005

arrival date search boundary (days)
-90, +90

*****
* 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 classical orbital elements *
* (heliocentric, Earth mean 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 (non-dimensional)
0.517491

orbital inclination (degrees)
10.5301

argument of perihelion (degrees)
178.8390

longitude of the ascending node (degrees)
68.9734

*****
* 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.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
-----
1

*****
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

```

Here's the optimal solution for this example including the verification of the *AMA_OC* solution.

```

program ipto_ocs
=====

flyby trajectory

minimize launch delta-v

DE421 ephemeris

DEPARTURE CONDITIONS
=====

calendar date      01/10/2005

TDB time           08:45:26.84

TDB julian date    2453380.86489403

heliocentric delta-v vector and magnitude
(Earth mean ecliptic and equinox of J2000)
-----

launch delta-vx    -2971.11698008735    meters/second
launch delta-vy    -1191.71198647956    meters/second
launch delta-vz    -335.134462946394    meters/second

launch delta-v     3218.69984253530    meters/second

```

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

```
-----
right ascension    197.907324746402    degrees
declination        -14.0521359432754    degrees
C3                  10.3600286763368      (km/sec)**2
```

heliocentric orbital elements and state vector of the departure planet
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1000965820D+01  0.1761052573D-01  0.7749154801D-03  0.3176116281D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.1455924113D+03  0.6945172026D+01  0.3245568001D+03  0.3657861848D+03

      rx (km)       ry (km)               rz (km)               rmag (km)
-.5067923758D+08  0.1381198216D+09  -.1153891645D+04  0.1471239962D+09

      vx (kps)      vy (kps)              vz (kps)              vmag (kps)
-.2846316247D+02  -.1037623019D+02  0.3333146047D-03  0.3029550743D+02
```

heliocentric orbital elements and state vector of the spacecraft
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.1300685590D+01  0.2438903757D+00  0.5726808349D+00  0.1803234359D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.2901042542D+03  0.3597215236D+03  0.1800449595D+03  0.5418223555D+03

      rx (km)       ry (km)               rz (km)               rmag (km)
-.5067923759D+08  0.1381198216D+09  -.1153891645D+04  0.1471239962D+09

      vx (kps)      vy (kps)              vz (kps)              vmag (kps)
-.3143427945D+02  -.1156794217D+02  -.3348011483D+00  0.3349691482D+02
```

ARRIVAL CONDITIONS
 =====

```
calendar date      07/10/2005
TDB time           02:21:44.73
TDB julian date    2453561.59843438
```

heliocentric orbital elements and state vector of the planet/asteroid/comet
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.3121531412D+01  0.5174910000D+00  0.1053010000D+02  0.1788390000D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.6897340000D+02  0.3140665417D+01  0.1819796654D+03  0.2014419845D+04

      rx (km)       ry (km)               rz (km)               rmag (km)
-.7369140595D+08  -.2130455810D+09  -.1423200348D+07  0.2254348429D+09

      vx (kps)      vy (kps)              vz (kps)              vmag (kps)
0.2759316336D+02  -.1009890910D+02  -.5461105864D+01  0.2988635652D+02
```

heliocentric orbital elements and state vector of the spacecraft
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.3501454000D+01  0.5700007349D+00  0.2868171372D+01  0.1832730028D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.6368061244D+02  0.3975119825D+01  0.1872481226D+03  0.2393156305D+04

      rx (km)       ry (km)                rz (km)               rmag (km)
-0.7369140595D+08 -0.2130455810D+09  -0.1423200348D+07  0.2254348429D+09

      vx (kps)      vy (kps)               vz (kps)              vmag (kps)
0.2843623711D+02  -0.1063584982D+02  -0.1513252337D+01  0.3039787517D+02

total delta-v      3218.69984253530      meters/second
transfer time      180.733540346846      days
```

```
=====
verification of optimal control solution
=====
```

heliocentric orbital elements and state vector of the spacecraft at arrival
 (Earth mean ecliptic and equinox of J2000)

```
-----
      sma (au)      eccentricity      inclination (deg)      argper (deg)
0.3501453652D+01  0.5700006990D+00  0.2868171382D+01  0.1832730062D+03

      raan (deg)    true anomaly (deg)    arglat (deg)          period (days)
0.6368061198D+02  0.3975116501D+01  0.1872481227D+03  0.2393155948D+04

      rx (km)       ry (km)                rz (km)               rmag (km)
-0.7369140625D+08 -0.2130455767D+09  -0.1423200336D+07  0.2254348390D+09

      vx (kps)      vy (kps)               vz (kps)              vmag (kps)
0.2843623719D+02  -0.1063584937D+02  -0.1513252335D+01  0.3039787509D+02
```

final heliocentric position vector and magnitude errors

```
delta rx      -0.301263228058815      kilometers
delta ry      4.24355044960976      kilometers
delta rz      1.195031986571848E-002 kilometers
delta rmag    4.25424761398634      kilometers
```

Here's the graphics display of the interplanetary transfer trajectory along with the planet and comet heliocentric orbits. The transfer trajectory is red and the scales are in astronomical units. For this example we can see that encounter with Tempel 1 occurs near perihelion of the comet's orbit.

Ballistic Interplanetary Trajectory Analysis

Earth-to-Tempel 1 Minimum Launch Delta-V

