A MATLAB Script for Ballistic Interplanetary Trajectory Design and Optimization

This document describes a MATLAB script called `ipto_snopt` that can be used to design and optimize "patched conic" interplanetary trajectories between any two planets of our solar system. It can also be used to find trajectories between a planet and an asteroid or comet. A patched-conic trajectory ignores the gravitational effect of both the launch and arrivals planets on the heliocentric trajectory. This technique involves the solution of Lambert's problem relative to the Sun. Patched-conic trajectories are suitable for preliminary mission design.

Typical user interaction

The following is typical user interaction with this MATLAB application. This example is an Earth-to-Mars mission that minimizes the launch delta-v. The user inputs for this example are in bold font.

```
departure conditions - start date

please input the calendar date
(1 <= month <= 12, 1 <= day <= 31, year = all digits!)
? 6,1,2003

please input the departure date search boundary in days
? 30


arrival conditions - start date

please input the calendar date
(1 <= month <= 12, 1 <= day <= 31, year = all digits!)
? 12,1,2003

please input the arrival date search boundary in days
? 30

 celestial body menu

  <1>  Mercury
  <2>  Venus
  <3>  Earth
  <4>  Mars
  <5>  Jupiter
  <6>  Saturn
  <7>  Uranus
  <8>  Neptune
  <9>  Pluto
  <10> asteroid/comet

please select the departure celestial body
? 3

 celestial body menu

  <1>  Mercury
  <2>  Venus
  <3>  Earth
  <4>  Mars
```

```
  <5>  Jupiter
  <6>  Saturn
  <7>  Uranus
  <8>  Neptune
  <9>  Pluto
  <10> asteroid/comet

please select the arrival celestial body
? 4

      optimization menu

 <1> minimize departure delta-v

 <2> minimize arrival delta-v

 <3> minimize total delta-v

 <4> no optimization

 selection (1, 2 or 3)
? 3
```

If the analyst selects an asteroid or comet as the launch or arrival body, the software will prompt the user for the name of a data file containing the orbital elements of the object. The orbital elements of an asteroid or comet relative to the ecliptic and equinox of J2000 coordinate system must be provided by the user. The following is a typical data file for the comet Tempel 1. Do not change the number of lines of information in these data files. The data values are in bold.

```
*********************************************
* 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
```

Optimal solution and trajectory graphics display

This section summarizes the program output for this example. The information provided by the software includes the heliocentric orbital elements of the initial orbit, transfer trajectory and final mission orbit in the J2000 mean ecliptic and equinox coordinate system. These numerical results also include the characteristics of the launch hyperbola.

```
          program ipto_snopt

        minimize total delta-v

 departure celestial body     Earth

 departure calendar date      06-Jun-2003
 departure universal time     08:17:20.581

 departure julian date        2452796.8454


 arrival celestial body       Mars

 arrival calendar date        27-Dec-2003
 arrival universal time       17:03:45.063

 arrival julian date          2453001.2109

 transfer time                  204.3656  days

 heliocentric orbital elements of the initial orbit prior to the first maneuver
 (mean ecliptic and equinox of J2000)

       sma (km)        eccentricity     inclination (deg)    argper (deg)
  1.4963256152e+008  1.6337778750e-002  3.2747257174e-004  2.7195703706e+002

      raan (deg)     true anomaly (deg)   arglat (deg)       period (days)
  1.9020118456e+002  1.5304069737e+002  6.4997734434e+001  3.6538395692e+002


 heliocentric orbital elements of the transfer orbit after the first maneuver
 (mean ecliptic and equinox of J2000)

       sma (km)        eccentricity     inclination (deg)    argper (deg)
  1.8842763117e+008  1.9438220509e-001  7.1019070378e-002  1.7893291901e+002

      raan (deg)     true anomaly (deg)   arglat (deg)       period (days)
  7.5438356029e+001  8.2764376688e-001  1.7976056278e+002  5.1632980036e+002


 heliocentric orbital elements of the transfer orbit prior to the second maneuver
 (mean ecliptic and equinox of J2000)

       sma (km)        eccentricity     inclination (deg)    argper (deg)
  1.8842763117e+008  1.9438220509e-001  7.1019070378e-002  1.7893291901e+002

      raan (deg)     true anomaly (deg)   arglat (deg)       period (days)
  7.5438356029e+001  1.5417471526e+002  3.3310763427e+002  5.1632980036e+002


 heliocentric orbital elements of the final orbit after the second maneuver
 (mean ecliptic and equinox of J2000)
```

```
      sma (km)        eccentricity     inclination (deg)    argper (deg)
 2.2793906443e+008  9.3540802491e-002  1.8493720042e+000  2.8651708377e+002

     raan (deg)     true anomaly (deg)   arglat (deg)       period (days)
 4.9540923322e+001  7.2487482565e+001  3.5900456634e+002  6.8697107418e+002


departure delta-v and energy requirements
(mean equator and equinox of J2000)

x-component of delta-v       2900.620665  meters/second
y-component of delta-v       -549.963086  meters/second
z-component of delta-v       -282.170562  meters/second

delta-v magnitude            2965.751147  meters/second

energy                          8.795680  km^2/sec^2

asymptote right ascension     349.264051  degrees
asymptote declination          -5.459552  degrees


arrival delta-v and energy requirements
(mean equator and equinox of J2000)

x-component of delta-v      -2021.548322  meters/second
y-component of delta-v       1170.832247  meters/second
z-component of delta-v       1357.142840  meters/second

delta-v magnitude            2701.729531  meters/second

energy                          7.299342  km^2/sec^2

asymptote right ascension     149.921608  degrees
asymptote declination          30.153856  degrees


total delta-v                5667.480678  meters/second

total energy                   16.095022  km^2/sec^2
```

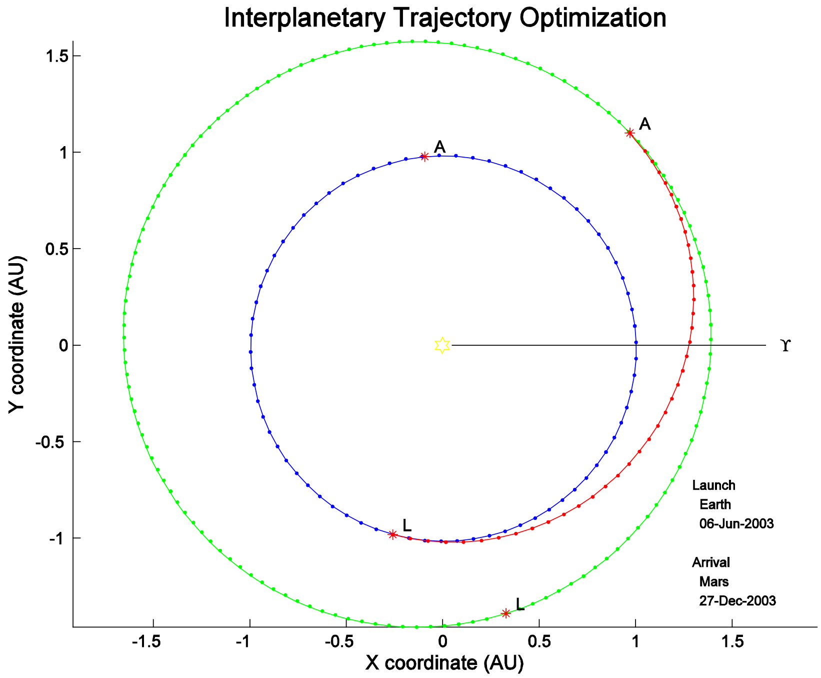
After the solution is displayed, the software will ask the user if he or she would like to create a graphics display of the transfer trajectory with the following prompt:

```
would you like to plot this trajectory (y = yes, n = no)
?
```

If the user's response is `y` for yes, the script will request a plot step size with

```
please input the plot step size (days)
?
```

The following is a typical graphics display created with this MATLAB script. The plot is a *north ecliptic* view where we are looking down on the ecliptic plane from the north celestial pole. The vernal equinox direction is the labeled line pointing to the right, the launch body is labeled with an `L` and the arrival body is labeled with an `A`. The location of the launch and arrival celestial bodies at the launch time is marked with an asterisk. The initial orbit trace is blue, the transfer trajectory is red and the final orbit trace is green.



Technical Discussion

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_snopt` software uses the user's initial guess for the launch and arrival dates.

Lambert's Problem

Lambert's problem is concerned with the determination of an orbit that passes between two positions within a specified time-of-flight. This classic astrodynamic problem is also known as the orbital two-point boundary value problem (TPBVP).

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. This relationship can be stated as follows:

$$tof = tof\left(r_i + r_f, c, a\right)$$

From the following form of Kepler's equation

$$t - t_0 = \sqrt{\frac{a^3}{\mu}}\left(E - e\sin E\right)$$

we can write

$$t = \sqrt{\frac{a^3}{\mu}}\left[E - E_0 - e\left(\sin E - \sin E_0\right)\right]$$

where E is the eccentric anomaly associated with radius r, E_0 is the eccentric anomaly at r_0, and $t = 0$ when $r = r_0$.

At this point we need to introduce the following trigonometric sun and difference identities:

$$\sin\alpha - \sin\beta = 2\sin\frac{\alpha-\beta}{2}\cos\frac{a+\beta}{2}$$

$$\cos\alpha - \cos\beta = -2\sin\frac{\alpha-\beta}{2}\sin\frac{a+\beta}{2}$$

$$\cos\alpha + \cos\beta = 2\cos\frac{\alpha-\beta}{2}\cos\frac{a+\beta}{2}$$

If we let $E = \alpha$ and $E_0 = \beta$ and substitute the first trig identity into the second equation above, we have the following equation:

$$t = \sqrt{\frac{a^3}{\mu}}\left\{E - E_0 - 2\sin\frac{E-E_0}{2}\left(e\cos\frac{E+E_0}{2}\right)\right\}$$

With the two substitutions given by

$$e\cos\frac{E+E_0}{2} = \cos\frac{\alpha+\beta}{2}$$

$$\sin\frac{E-E_0}{2} = \sin\frac{\alpha-\beta}{2}$$

the time equation becomes

$$t = \sqrt{\frac{a^3}{\mu}\left\{\left(\alpha-\beta\right) - 2\sin\frac{\alpha-\beta}{2}\cos\frac{\alpha+\beta}{2}\right\}}$$

From the elliptic relationships given by

$$r = a\left(1 - e\cos E\right)$$

$$x = a\left(\cos E - e\right)$$

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

and some more manipulation, we have the following equations:

$$\cos\alpha = \left(1 - \frac{r + r_0}{2a}\right) - \frac{c}{2a} = 1 - \frac{r + r_0 + c}{2a} = 1 - \frac{s}{a}$$

$$\sin\beta = \left(1 - \frac{r + r_0}{2a}\right) + \frac{c}{2a} = 1 - \frac{r + r_0 - c}{2a} = 1 - \frac{s - c}{a}$$

This part of the derivation makes use of the following three relationships:

$$\cos\frac{\alpha - \beta}{2}\cos\frac{\alpha + \beta}{2} = 1 - \frac{r + r_0}{2}$$

$$\sin\frac{\alpha - \beta}{2}\sin\frac{\alpha + \beta}{2} = \sin\frac{E - E_0}{2}\sqrt{1 - \left(e\cos\frac{E + E_0}{2}\right)^2}$$

$$\left(\sin\frac{\alpha - \beta}{2}\sin\frac{\alpha + \beta}{2}\right)^2 = \left(\frac{x - x_0}{2a}\right)^2 + \left(\frac{y - y_0}{2a}\right)^2 = \left(\frac{c}{2a}\right)^2$$

With the use of the half angle formulas given by

$$\sin\frac{\alpha}{2} = \sqrt{\frac{s}{2a}} \qquad \sin\frac{\beta}{2} = \sqrt{\frac{s - c}{2a}}$$

and several additional substitutions, we have the time-of-flight form of Lambert's theorem

$$t = \sqrt{\frac{a^3}{\mu}}\left[\left(\alpha - \beta\right) - \left(\sin\alpha - \sin\beta\right)\right]$$

A discussion about the angles ∟ and *b* can be found in "Geometrical Interpretation of the Angles *a* and *b* in Lambert's Problem" by J. E. Prussing, AIAA *Journal of Guidance and Control*, Volume 2, Number 5, Sept.-Oct. 1979, pages 442-443.

The algorithm used in this MATLAB script 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.

The planetary ephemeris used in this computer program is based on the JPL DE421 ephemeris.

The orbital elements of an asteroid or comet relative to the ecliptic and equinox of J2000 coordinate system must be provided by the user. These elements can be obtained from the JPL Small-Body Database Browser (http://ssd.jpl.nasa.gov/sbdb.cgi), the MPC database at Harvard (http://cfa-www.harvard.edu) or the Bureau of Longitudes in Paris (http://www.bdl.fr).

These orbital elements consist of the following items:

- 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:

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

where μ_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 MATLAB script 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

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

The fundamental equation we want to solve is

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

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:

$$\left| f(E) \right| \le \varepsilon$$

where ε is the convergence tolerance. This tolerance is hardwired in the software to ε =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 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:

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

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

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

The $\Delta V's$ required at launch and arrival are simply the differences between the velocity on the transfer trajectory determined by the solution of Lambert's problem and the heliocentric velocities of the two celestial bodies. If we treat each celestial body as a point mass and assume *impulsive* maneuvers, the *body-centered* magnitude and direction of the required maneuvers are given by the two vector equations:

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

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

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
$\mathbf{V}_{B_L}$ = heliocentric velocity vector of the celestial body at launch
$\mathbf{V}_{B_A}$ = heliocentric velocity vector of the celestial body at arrival

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 celestial body at an *infinite* distance from the body. 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 <u>twice</u> the orbital energy per unit mass (the specific orbital energy).

The orientation of the departure and arrival hyperbolas is specified in terms of the right ascension and declination of the asymptote. These coordinates can be calculated using the components of the V_∞ velocity vector.

The right ascension of the asymptote is determined from

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

and the geocentric declination of the asymptote is given by

$$\delta = 90^0 - \cos^{-1}\left(\Delta \hat{V}_z\right)$$

where $\Delta \hat{V}_z$ is z-component of the unit ΔV vector.

In this script the heliocentric planetary coordinates and therefore the ΔV vectors are computed in the J2000 ecliptic and equinox 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, $\Delta \mathbf{V}_{eq}$ is the delta-velocity vector in the equatorial frame and ε is the mean obliquity of the ecliptic at the departure or arrival date.

In the terminology of numerical optimization, this MATLAB script treats the launch and arrival dates as *control or optimization variables* and attempts to minimize the launch, arrival or sum of launch and arrival <u>scalar</u> $\Delta V's$. The scalar magnitude of the selected ΔV is called *the objective function* or *the performance index*.

The software can solve for the following types of optimized interplanetary space missions:

- minimum launch ΔV
- minimum arrival ΔV
- minimum total ΔV

The software will ask the user for an initial guess for the launch and arrival calendar dates as well as the launch and arrival celestial bodies. The software will also ask the user for a search boundary, in days, on the launch and arrival dates. The algorithm will restrict its search for the optimum launch date D_L and arrival date D_A as follows:

$$D_{L_g} - \Delta D_L \le D_L \le D_{L_g} + \Delta D_L$$

$$D_{A_g} - \Delta D_A \le D_A \le \Delta D_A + \Delta D_A$$

where D_{L_g} and D_{A_g} are the user's initial guess for launch and arrival dates, and ΔD_L, ΔD_A, are the user-specified search boundaries for the launch and arrival dates, respectively.

SNOPT algorithm implementation

This section provides details about the part of the `ipto_snopt` MATLAB script that solves this nonlinear programming (NLP) problem using the SNOPT 6.0 algorithm. In this classic trajectory optimization problem, the launch and arrival calendar dates are the *control variables* and the scalar ΔV computed by the solution of Lambert's problem is the *objective function* or *performance index*.

MATLAB versions of SNOPT 6.0 for several computer platforms can be found at Professor Philip Gill's web site which is located at http://scicomp.ucsd.edu/~peg/.

The SNOPT algorithm requires an initial guess for the control variables. For this problem they are given by

```
xg(1) = jdate1 - jdate0;

xg(2) = jdate2 - jdate0;

xg = xg';
```

where `jdate1` and `jdate2` are the initial user-provided launch and arrival date guesses, and `jdate0` is a reference Julian date equal to `2451544.5` (January 1, 2000). This offset value is used to *scale* the control variables.

The algorithm also requires lower and upper bounds for the control variables. These are determined from the initial guesses and user-defined search boundaries as follows:

```
% bounds on control variables

xlwr(1) = xg(1) - ddays1;
xupr(1) = xg(1) + ddays1;

xlwr(2) = xg(2) - ddays2;
xupr(2) = xg(2) + ddays2;

xlwr = xlwr';
xupr = xupr';

xlwr = xlwr';
xupr = xupr';
```

where `ddays1` and `ddays2` are the user-defined launch and arrival search boundaries, respectively.

The algorithm also requires lower and upper bounds on the objective function. For this problem these bounds are given by

```
% bounds on objective function

flow(1) = 0.0d0;
fupp(1) = +Inf;
```

The actual call to the SNOPT MATLAB interface function is as follows

```
[x, f, inform, xmul, fmul] = snopt(xg, xlwr, xupr, flow, fupp, 'iptofunc');
```

where `iptofunc` is the name of the MATLAB function that solves Lambert's problem and computes the current value of the objective function.

The `ipto_snopt` script will also read an SNOPT SPECS file. For this example the contents of this file are as follows:

```
Begin SNOPT options
    minor iterations limit       1000
    derivative option            0
    major optimality tolerance   1.0d-6
    solution                     Yes
End SNOPT options
```

Please consult the SNOPT documentation for a complete explanation of the SPECS file. A PDF and Postscript version of the SNOPT user's manual is also available at Professor Gill's website.

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.

"User's Guide for SNOPT Version 6, A Fortran Package for Large-Scale Nonlinear Programming", Philip E. Gill, Walter Murray and Michael A. Saunders, December 2002.

"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.