OITS - IMPORTANT

So far there is very little input validation with this software.

SETUP:

1) To run this software you will in addition need the SPICE binary kernel file de430.bsp from the following link:
https://naif.jpl.nasa.gov/pub/naif/pds/wgc/kernels/spk/
It must go in the SPICE directory provided with this software. It exceeds the 100MB limit which is why it is not included in this repository.

OPERATION:

1) Type 'run_OITS' at the MATLAB command line
2) Click on 'NEW MISSION'
3) Select your sequence of Planets, starting from the Home Planet and ending with Destination Planet. Use the Planet Barycentres when selecting Planets.
4) If choosing an 'INTERMEDIATE POINT', a maximum of one is advisable. Also see NOTE below. Do not use at all if beginner. ('FIXED POINT' should not be used yet).
5) Click on ''QUIT SELECTION'
6) Click on 'SET OPTIMIZER DETAILS'
7) Set Time Bounds and Initial Guess. Note for first Planet, use Launch Date in indicated format, for subsequent Planets use Flight Time in days. Note Initial Guess must satisfy Time Bounds.
8) Set Minimum Periapsis Altitude in km if desired (defaults to 0). Note this will be ignored for first and last planet.
9) Set Minimum Perihelia for Transfer in AU if desired (defaults to 0).
10) Set Maximum Duration of Mission in years if desired (defaults to no Maximum Duration).
11) Set Run Time for Global Optimization in minutes if desired (defaults to 1 minute -for greater number of Planets this needs to be increased).
12) Select Rendezvous with Destination if desired (defaults to flyby).
13) Prograde only is recommended (default).

14) Close Window.
15) Click on ‘RUN OPTIMIZATION’. To stop Optimization at any stage press Control-C in MATLAB output window.
16) When finished click on ‘VIEW RESULTS’.
17) After viewing results, ‘SAVE MISSION’ if desired.

NOTE:
If ‘INTERMEDIATE POINT’ is selected remember to specify distance from Sun in AU in the Minimum Periapsis Altitude column when setting Optimizer Details.

THEORY

I am working on a definition file for OITS, a draft in pdf format is provided here. Adopting various assumptions, my Optimum Interplanetary Trajectory Software (OITS) can compute a unique interplanetary trajectory as a function of purely the times of encounter at each of the Solar System Objects visited (there are a user specified combination of ‘Nbody’ SSO’s altogether). Thus given an array of encounter times t_i, i= 1,2,… Nbody, a complete trajectory can be derived, complete in terms of:

1) the interplanetary trajectories connecting up each successive pair of SSO’s in turn – the sun is the gravitational attracting centre (inverse square field)
2) the encounter hyperbolae at each of the SSO’s in turn – the SSO in question is the gravitational attracting centre (inverse square field)

Thus a patched conic assumption is used where at any time the spacecraft is under the influence of either the sun or an SSO – not both at the same time.

In the above, solving 1 is the Lambert problem, otherwise known as the Gauss problem. Given two successive SSO’s, numbered j and j+1, and given the times t_j and t_{j+1}, then positions $\mathbf{r}_j$ and $\mathbf{r}_{j+1}$ and velocities $\mathbf{v}_j$ and $\mathbf{v}_{j+1}$ can be acquired by either:

a) use of the NASA SPICE toolkit which can be linked in with MATLAB code or
b) use of the orbital elements/epoch and a prediction problem solution by Universal Variables (see below).

The Lambert/Gauss problem is then solved by OITS using the 'Universal Variable Formulation' as detailed in '*Fundamentals of Astrodynamics'* by Bate, Mueller and White. A Newton iteration method was implemented to solve this problem.

Having solved problem 1, we are presented with problem 2. Taking an arbitrary SSO, numbered j, we now have the heliocentric velocity of arrival of the s/c at the SSO as $\mathbf{VARR_j}$ and the heliocentric departure velocity $\mathbf{VDEP_j}$, both of these are derived from 1 above. The SSO-centred velocities $\mathbf{VA_j}$ and $\mathbf{VD_j}$ respectively are then given by:

$$\mathbf{VA_j} = \mathbf{VARR_j} - \mathbf{v_j}$$

$$\mathbf{VD_j} = \mathbf{VDEP_j} - \mathbf{v_j}$$

Thus it is assumed that the encounter time is negligible compared to the duration of the interplanetary trajectories calculated in 1. Hence $\mathbf{VA_j}$ and $\mathbf{VD_j}$ are adopted as the SSO-centred hyperbolic excess for arrival and departure respectively.

It is further assumed that a change in velocity is applied at the periapsis point of the s/c w.r.t. the SSO, in plane and tangential to the velocity (i.e. perpendicular to the SSO-centred radius vector). With these assumptions, two coplanar connecting hyperbolae can be calculated, one for arrival up to periapsis and one from periapsis to departure with the extra velocity applied at periapsis. Refer to OITS Definition File Sections 1.6.6 and 1.6.7. A numerical solution is required and was implemented in OITS using a Newton Iteration.

This velocity is ΔV_j. The sum of these is

$$\Delta V_{TOT} = \Sigma \Delta V_j$$

This is the metric for the trajectory and the lower this value the more viable the trajectory.

ΔV_1 is the hyperbolic excess at SSO_1 .

ΔV_{Nbody} is either 0 or if rendezvous at target SSO is selected, equal to $|\mathbf{VARR_{Nbody}} - \mathbf{v_{Nbody}}|$.

NB, as well as specifying an SSO as an object to encounter along the trajectory, OITS provides the option of an 'Intermediate Point' with a user-specified radial distance from the origin of the ecliptic. Thus the polar angles of this point are variables which can be added to the optimization variables t_i.

As we have seen ΔV_{TOT} is a function of the time array/vector t_i, i= 1,2,… Nbody .

Stated simply, we have a function f as follows:

$$\Phi = f(t_1, t_2, t_3 \ldots . t_{Nbody})$$

Where Φ is equivalent to ΔV_{TOT} and f is essentially equivalent to solution of problems 1 and 2 above.

For each of the SSO encounters, j=2,3,..(Nbody-1), there will be a minimum periapsis radius (at least the radius of the Planet's equator). This provides us with the following inequality constraints:

$$g_j(t_1, t_2, t_3 \ldots . t_{Nbody}) \leq 0 \qquad j=2,3,\ldots(Nbody-1)$$

In addition there may be optional user-specified constraints – minimum perihelia and/or maximum total mission duration.

This is a Non-Linear Global Optimization Problem with Inequality Constraints. Various free NLGOP solvers are available on the internet and after some research it was found that 'NOMAD' was the most effective for the problem in question. NOMAD is available for MATLAB via the OPTI Toolbox site. Please find user guide in pdf format attached.