

Program itcm_soc

Interplanetary TCM Optimization with SOCS

This document is the user's manual for a Fortran computer program called `itcm_soc` that uses the Sparse Optimal Control Software (SOCS) object code library developed by Boeing (www.boeing.com/phantom/socs/) to solve the classic one impulse interplanetary trajectory correction maneuver (TCM) optimization problem. The software attempts to minimize the scalar magnitude of the TCM delta-v vector for user-defined final orbit conditions at Mars. With the additional of proper coordinate transformations, the software can also be used for other planets.

The important features of this scientific simulation are as follows:

- heliocentric, inertial cartesian equations of motion with point-mass planetary perturbations
- elliptical, non-coplanar planetary orbits
- JPL DE421 planetary ephemeris model
- B-plane coordinates of the encounter hyperbola

The final conditions at Mars can be determined using one of the following user-defined options;

- radius and orbital inclination
- user-defined B-plane coordinates
- grazing flyby; user-defined B-plane angle

SOCS 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 SOCS can be found in the book, "Practical Methods for Optimal Control Using Nonlinear Programming" by John. T. Betts, SIAM, 2001.

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

- main program that sets algorithm control parameters and calls the SOCS transcription/optimal control subroutine
- define problem definition and perform initialization related to scaling, lower and upper bounds, initial conditions, etc.
- evaluate the *right-hand-side* differential equations
- define and compute any point and path constraints
- display the optimal solution results

SOCS 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 collocation method and other important algorithm control parameters.

Typical Input File

The `itcm_soc`s 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 trajectory that begins at the Earth's sphere-of-influence (SOI) and ends at hyperbolic encounter with Mars. It can be found as `e2m1.in` in the software distribution.

In the following discussion the actual input file contents are in *courier* font and all explanations are in *times italic* font. Each data item within an input file is preceded by one or more lines of *annotation* text. Do not delete any of these annotation lines or 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.

```
*****
** interplanetary TCM trajectory optimization
** n-body heliocentric motion
** Earth-to-Mars data file - e2m1.in
** March 14, 2007
*****
```

The first program input is the julian date, on the TDB time scale, at which to perform the TCM.

```
TDB julian date of TCM
2452799.264399034436792d0
```

The next six inputs are the components of the spacecraft's heliocentric orbital elements prior to the actual trajectory correction maneuver. These coordinates are defined relative to the Earth mean equator and equinox of J2000 system (EME2000) at the time specified in the previous input.

```
*****
heliocentric EME2000 orbital elements prior to TCM
*****

semimajor axis (kilometers)
190725765.750D0

orbital eccentricity (non-dimensional)
0.204056802425D0

orbital inclination (degrees)
23.4926769446D0

argument of perihelion (degrees)
253.488459798D0

right ascension of the ascending node (degrees)
0.463121032996D0
```

true anomaly (degrees)
3.94861259377D0

The next three inputs are the user's initial guess for the components of the TCM delta-v vector. These values should be provided in the units of meters per second. If an initial guess is not available, the user should input 0 for each component.

initial guess and bounds for heliocentric TCM delta-v vector

x-component of TCM velocity vector (meters/second)
0.0

y-component of TCM velocity vector (meters/second)
0.0

z-component of TCM velocity vector (meters/second)
0.0

Lower and upper bounds for each component of the TCM delta-v vector are defined by the next two user inputs. The units for these two numbers are also meters per second.

lower bound for TCM delta-v components (meters/second)
-100.0

upper bound for TCM delta-v components (meters/second)
+100.0

The next integer input allows the user to specify the type of final orbit targeting at Mars.

final orbit targeting options

1 = user-defined radius and orbital inclination
2 = user-defined B-plane coordinates
3 = grazing flyby; user-defined b-plane angle

1

The next set of inputs defines the desired characteristics of the encounter hyperbola at Mars. The orbital inclination and B-plane coordinates are defined with respect to the Mars mean equator and equinox of date coordinate system.

final Mars-centered targets

periapsis radius (kilometers)
5000.0d0

orbital inclination (degrees)
60.0d0

user-defined b dot r target (kilometers)
-7889.908599155647607d0

user-defined b dot t target (kilometers)
4607.242716469171683d0

```
user-defined b-plane angle (degrees)
-60.0d0
```

The next two inputs set the root-finding and truncation error tolerances used by the numerical methods that calculate the initial guess, predict the time of entrance into the Martian sphere-of-influence, and the time of closest approach to Mars. Smaller values will improve the solution of the corresponding problem at the expense of longer computation times.

```
*****
root-finding and integration algorithm control
*****

root-finding tolerance
1.0d-8

RKF7(8) truncation error tolerance
1.0d-12
```

The next input is the user-defined value for the radius of the sphere-of-influence of Mars.

```
-----
radius of Mars sphere-of-influence (kilometers)
-----
150000.0
```

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 SOCS. 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 = SOCS-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 for a description of the information written to this file.

```
name of solution output file
e2m1.csv
```

The next series of program inputs are algorithm control options and parameters for the SOCS software. The first input is an integer that specifies the type of collocation method to use during the solution process. For most simulations, the separated trapezoidal method is recommended.

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

```

discretization/collocation method
-----
1 = trapezoidal
2 = separated Hermite-Simpson
3 = compressed Hermite-Simpson
4 = Runge-Kutta 4-stage
-----
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 SOCS 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 SOCS 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 SOCS 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 SOCS 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 and Trajectory Plots

The following is the program output created by the `itcm_soc`s simulation for this example. Please see Appendix B for additional details about this information.

```
-----
program itcm_soc
-----

input data file ==> e2m1.in

user-defined radius and orbital inclination

TCM delta-v vector and magnitude
(heliocentric EME2000)
-----

delta-vx          3.204870365873388 meters/second
delta-vy          20.009081087383386 meters/second
delta-vz          7.307702470364198E-003 meters/second

deltav           20.264120346712620 meters/second

time and conditions prior to TCM maneuver
(heliocentric EME2000)
-----

calendar date      June      8, 2003

TDB time           18:20:44.077

TDB Julian date    2452799.264399034436792
```

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.190725765750D+09	0.204056802425D+00	0.234926769446D+02	0.253488459798D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.463121032996D+00	0.394861259377D+01	0.257437072392D+03	0.757158581013D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.319331575699D+08	-.136207676243D+09	-.590899587841D+08	0.151867971768D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.316260608115D+02	-.655290820823D+01	-.295930905686D+01	0.324330978868D+02

time and conditions after TCM maneuver
(heliocentric EME2000)

calendar date June 8, 2003
TDB time 18:20:44.077
TDB Julian date 2452799.264399034436792

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.190709553333D+09	0.203958518176D+00	0.234957392553D+02	0.253649651358D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.497546517162D+00	0.375584940932D+01	0.257405500767D+03	0.757062041018D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.319331575699D+08	-.136207676243D+09	-.590899587841D+08	0.151867971768D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.316292656819D+02	-.653289912715D+01	-.295930174916D+01	0.324321859489D+02

time and conditions at Mars SOI
(heliocentric EME2000)

calendar date December 23, 2003
TDB time 15:13:35.094
TDB Julian date 2452997.134433957748115

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.187247635866D+09	0.200439438068D+00	0.234395610701D+02	0.252539281842D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.339350854369D+00	0.153437179708D+03	0.459764615493D+02	0.736541653589D+06
rx (km)	ry (km)	rz (km)	rmag (km)
0.151326013625D+09	0.145367065109D+09	0.626354599136D+08	0.218984809648D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.144372414508D+02	0.157415779910D+02	0.686185055906D+01	0.224347099703D+02

time and conditions at Mars SOI
(Mars-centered mean equator and equinox of date)

calendar date December 23, 2003
TDB time 15:13:35.094
TDB Julian date 2452997.134433957748115

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.584908602638D+04	0.185484147945D+01	0.600047735995D+02	0.114036396487D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105597284785D+03	0.240802282472D+03	0.354838678959D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.336699731451D+05	0.145704334142D+06	-.116867430299D+05	0.150000000140D+06
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.546514982343D+00	-.273212351525D+01	0.360695752280D+00	0.280949798971D+01

b-plane coordinates at Mars SOI
(Mars-centered mean equator and equinox of date)

b-magnitude 9137.382422371420034 kilometers
b dot r -7891.127181646455938
b dot t 4606.720019248247809
theta 300.275732745938967 degrees
vinf 2.705962663929141 km/sec
r-periapsis 5000.041352201649715 kilometers
decl-asy 7.430788560557559 degrees
rasc-asy 281.279645433629867 degrees

flight path angle -86.636466750839745 degrees

time and conditions at Mars closest approach
(Mars-centered mean equator and equinox of date)

calendar date December 23, 2003
TDB time 00:00:51.729
TDB Julian date 2452996.500598715152591

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.585451822813D+04	0.185404123244D+01	0.600000001673D+02	0.114030007231D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105570857975D+03	0.931266410733D-11	0.114030007231D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.165298729450D+04	-.257426385273D+04	0.395484490839D+04	0.499999996287D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.218195356445D+01	-.407988592503D+01	-.174367427815D+01	0.494436957630D+01

b-plane coordinates at Mars closest approach
(Mars-centered mean equator and equinox of date)

b-magnitude 9140.305327248879621 kilometers
b dot r -7893.147626973563092
b dot t 4609.056521039420659
theta 300.281996161573886 degrees
vinf 2.704706988748581 km/sec
r-periapsis 4999.999962873018376 kilometers
decl-asy 7.449618441445357 degrees
rasc-asy 281.241297808562138 degrees

flight path angle 6.057991269011913E-012 degrees

time and conditions of Mars at closest approach
(heliocentric EME2000)

calendar date December 23, 2003

TDB time 00:00:51.729

TDB Julian date 2452996.500598715152591

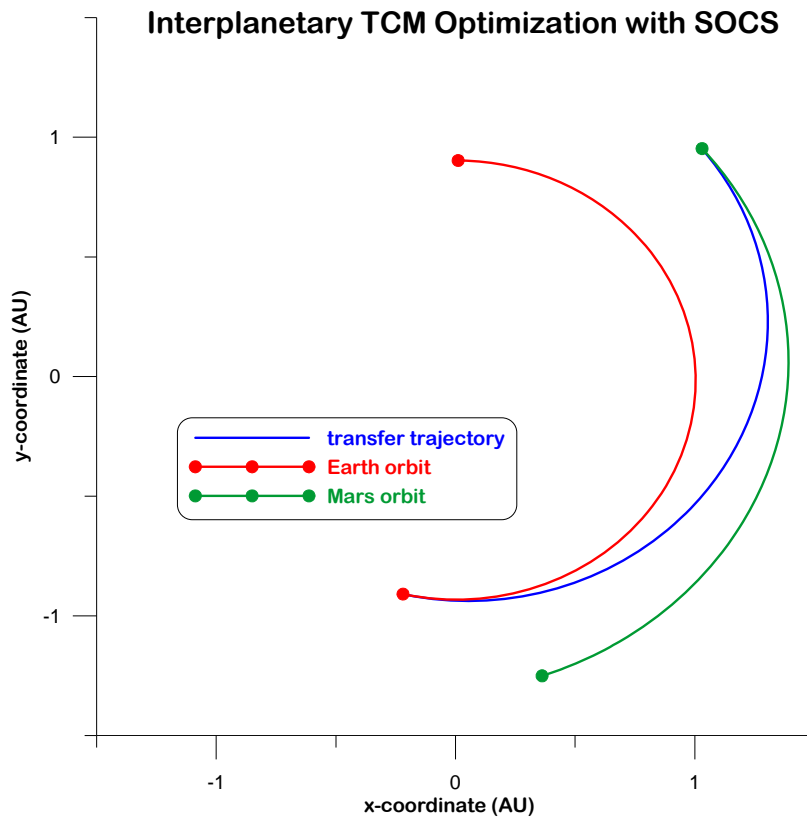
sma (au)	eccentricity	inclination (deg)	argper (deg)
0.152368095644D+01	0.935424560763D-01	0.246772247412D+02	0.332979394140D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.337165765326D+01	0.698324561144D+02	0.428118502543D+02	0.540647074355D-06
rx (km)	ry (km)	rz (km)	rmag (km)
0.152344735218D+09	0.144377942194D+09	0.621053617453D+08	0.218885779571D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.164717629596D+02	0.170512498608D+02	0.826603444180D+01	0.251075969022D+02

transfer time 197.236199680715799 days

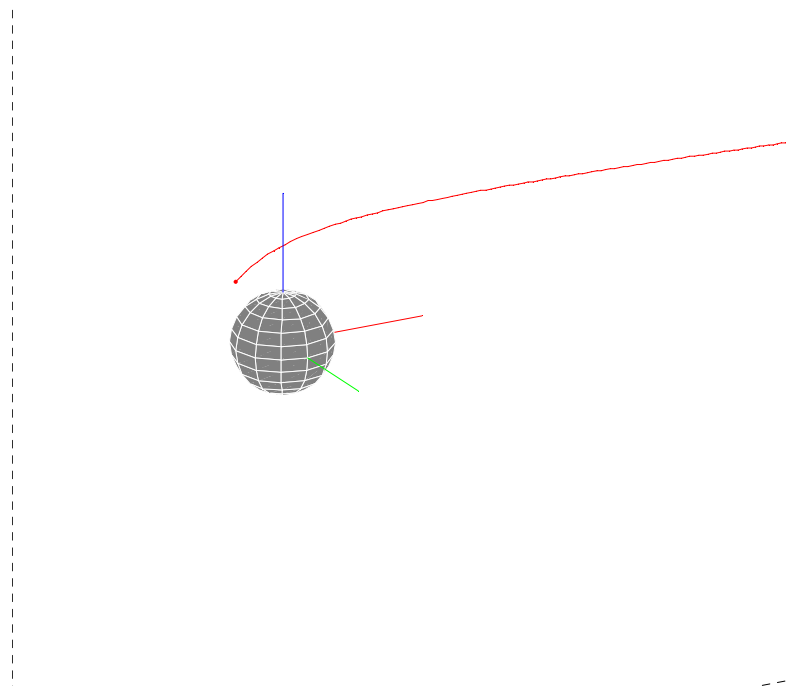
The `itcm_soc`s software will create two comma-separated-variable (csv) output files. The first file has the name specified by the user and contains the heliocentric state vectors of the spacecraft and the Earth and Mars. The second file is called `soiorb.csv` and contains the Mars-centered state vector at the sphere-of-influence.

The `itcm_soc`s software package also includes a MATLAB script called `mplot.m` that can be used to create trajectory graphic displays using the `soiorb.csv` data file. The interactive graphic features of MATLAB allow the user to rotate and zoom the displays. These capabilities allow the user to interactively find the best viewpoint as well as verify basic orbital geometry of the encounter trajectory at Mars.

The following is the heliocentric 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.



This next plot is a “zoomed” view of the encounter trajectory at Mars. This display is also labeled with a Mars centered, inertial coordinate system. The x-axis of this system is red, the y-axis green and the z-axis blue.



Problem setup for SOCS

This section provides additional details about the software implementation. For good scaling during the optimization, the time unit used in all heliocentric calculations is days, position is expressed in astronomical units, and the velocity and TCM delta-v unit is astronomical units per day.

(2) Performance index – minimize TCM delta-v

The objective function or performance index J for this simulation is the scalar magnitude of the TCM delta-v vector. For this classic trajectory optimization problem, this index is simply

$$J = \Delta V$$

The value of the `maxmin` indicator in SOCS tells the software whether the user is minimizing or maximizing the performance index.

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

The optimal solution must satisfy the following state vector boundary conditions (equality constraints) at the initial time defined by the user:

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

where $\mathbf{r}_{s/c}^-$ and $\mathbf{v}_{s/c}^-$ are the heliocentric position and velocity vectors of the spacecraft prior to the TCM maneuver time t_{TCM} , $\mathbf{r}_{s/c}^+$ and $\mathbf{v}_{s/c}^+$ are the heliocentric position and velocity vectors of the spacecraft after the TCM delta-v is applied, and $\Delta \mathbf{v}$ is the *impulsive* TCM delta-v vector.

(4) Point functions – SOI distance and final orbit targets at encounter

At the sphere-of-influence of Mars, the point function enforced by the software is given by

$$r_{SOI_p} - r_{SOI} = 0$$

Targeting to a Mars-centered periapsis radius and orbital inclination

For user-defined periapsis radius and orbital inclination targets at Mars, the following point functions or equality constraints are enforced

$$\begin{aligned} r_p - r_{ca} &= 0 \\ \cos i - \hat{\mathbf{h}}_z &= 0 \end{aligned}$$

where r_p and i are the user-defined periapsis radius and Mars centered orbital inclination of the encounter hyperbola, respectively. In the second equation, $\hat{\mathbf{h}}_z$ is the z-component of the predicted unit angular momentum vector. These orbital elements are determined from the spacecraft's state vector at closest approach to Mars.

Targeting to a Mars-centered grazing flyby

The general expression for the periapsis radius of an encounter hyperbola at Mars is given by

$$\tilde{r}_p = \frac{1}{\tilde{v}_\infty^2} \left(\sqrt{1 + \tilde{b}_\infty^2 \tilde{v}_\infty^4} - 1 \right)$$

where the *normalized* quantities are

$$\begin{aligned} \tilde{r}_p &= \text{normalized periapsis radius} = r_p / r_m \\ \tilde{b}_\infty &= \text{normalized b-plane magnitude} = b_\infty / r_m \\ \tilde{v}_\infty &= \text{normalized v-infinity speed} = v_\infty / v_{lc} \\ v_{lc} &= \text{local circular speed at Mars} = \sqrt{\mu_m / r_m} \\ r_m &= \text{radius of Mars} \\ \mu_m &= \text{gravitational constant of Mars} \end{aligned}$$

For a grazing flyby, $\tilde{r}_p = 1$ and the normalized B-plane distance is equal to

$$\tilde{b}_\infty = \sqrt{1 + \frac{2}{\tilde{v}_\infty^2}}$$

The required B-plane equality constraints are computed from

$$\mathbf{B} \cdot \mathbf{T} = b_\infty \cos \theta$$

$$\mathbf{B} \cdot \mathbf{R} = b_\infty \sin \theta$$

where θ is the user-defined B-plane angle of the grazing trajectory. Please note that the B-plane angle is measured positive clockwise from the \mathbf{T} axis of the B-plane coordinate system. The two equality constraints for this program option are simply the difference between the predicted and required $\mathbf{B} \cdot \mathbf{T}$ and $\mathbf{B} \cdot \mathbf{R}$ components.

Targeting to user-defined B-plane coordinates

For this program option, the two equality constraints are simply the difference between the predicted and the user-defined $\mathbf{B} \cdot \mathbf{T}$ and $\mathbf{B} \cdot \mathbf{R}$ components. The B-plane constraints for this and the previous option are scaled by dividing by the equatorial radius of Mars.

The components of the impulsive TCM delta- v are the optimization parameters used by SOCS to solve this trajectory problem.

Initial Guess

An initial guess for the heliocentric transfer trajectory is created by numerically integrating the heliocentric equations of motion using the user's input for the initial state of the spacecraft and the initial guess for the TCM delta- v vector. The position and velocity vectors at each grid point are determined by SOCS by setting the initial guess option `INIT(1) = 6` with `INIT(2) = 2`. The final time is estimated by searching for the instance at which the spacecraft reaches the sphere-of-influence of Mars. The numerical technique used to find the SOI consists of Brent's one-dimensional root-finder imbedded within a Runge-Kutta-Fehlberg 7(8) integrator. If an initial guess for the TCM maneuver is not available, a larger SOI radius should be used.

Technical Discussion

This section provides additional details about the numerical algorithms used in this computer program. The numerical methods discussed here include; propagating the spacecraft's heliocentric trajectory, B-plane coordinates and the Mars-centered coordinate transformation.

The fundamental coordinate system used in this simulation is Earth mean equator and equinox of J2000 (EME2000) and the fundamental time system is Barycentric Dynamical Time (TDB). All units are metric.

Heliocentric equations of motion

The general, second-order vector equation of motion subject to *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 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 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 SOCS 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. The equations described here are coded in the right-hand-side subroutine required by SOCS.

In this computer program the heliocentric coordinates of the planets are based on the JPL Development Ephemeris DE421. These coordinates are provided in the Earth mean equator and equinox of J2000 coordinate system (EME2000).

Time from the Martian SOI to closest approach

The elapsed time from the Martian SOI until closest approach to Mars is determined by an algorithm that includes a one-dimensional root-finder embedded with a Runge-Kutta-Fehlberg 7(8) numerical integration method. This technique searches for the time at which the Mars-centered flight path angle of the spacecraft is zero. This mission constraint is computed as follows

$$\gamma = \sin^{-1} \left(\frac{\mathbf{r} \cdot \mathbf{v}}{|\mathbf{r} \cdot \mathbf{v}|} \right)$$

where \mathbf{r} and \mathbf{v} are the Mars-centered position and velocity vectors, respectively.

The RKF78 method numerically solves the following system of six first-order differential equations

$$\dot{y}_1 = v_x = y_4 \quad \dot{y}_2 = v_y = y_5 \quad \dot{y}_3 = v_z = y_6$$

$$\dot{y}_4 = -\mu \frac{r_x}{r^3} \left\{ 1 + \frac{3}{2} \frac{J_2 r_{eq}^2}{r^2} \left(1 - \frac{5r_z^2}{r^2} \right) \right\} + a_{s_x}$$

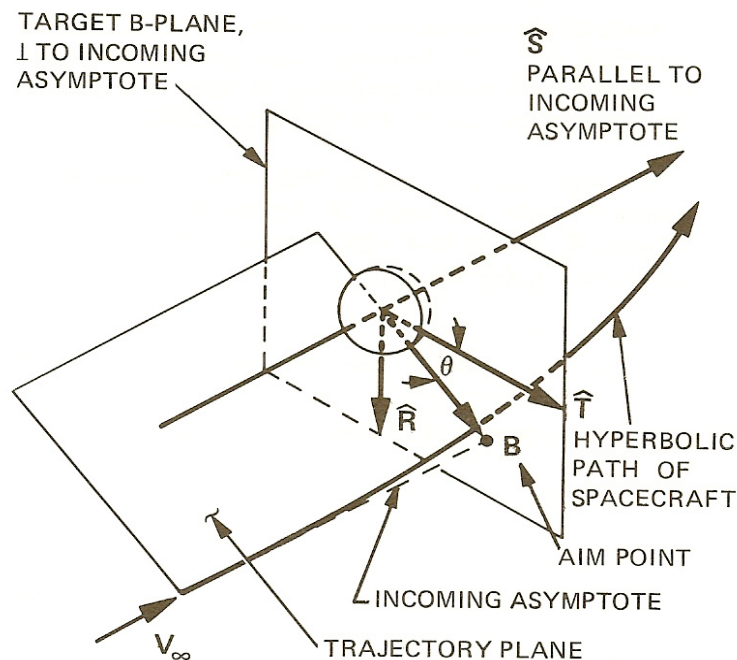
$$\dot{y}_5 = -\mu \frac{r_y}{r^3} \left\{ 1 + \frac{3}{2} \frac{J_2 r_{eq}^2}{r^2} \left(1 - \frac{5r_z^2}{r^2} \right) \right\} + a_{s_y}$$

$$\dot{y}_6 = -\mu \frac{r_z}{r^3} \left\{ 1 + \frac{3}{2} \frac{J_2 r_{eq}^2}{r^2} \left(3 - \frac{5r_z^2}{r^2} \right) \right\} + a_{s_z}$$

In these equations, μ and r_{eq} are the gravitational constant and equatorial radius of Mars, respectively, J_2 is the oblateness gravity coefficient, and the Mars-centered acceleration vector of the sun is represented by a_{s_x} , a_{s_y} and a_{s_z} .

The B-plane

The derivation of B-plane coordinates is described in the classic JPL reports, "A Method of Describing Miss Distances for Lunar and Interplanetary Trajectories" and "Some Orbital Elements Useful in Space Trajectory Calculations", both by William Kizner. The following diagram illustrates the fundamental geometry of the B-plane coordinate system.



The arrival asymptote unit vector $\hat{\mathbf{S}}$ is given by

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

where δ_{∞} and α_{∞} are the declination and right ascension of the asymptote of the incoming hyperbola.

The following computational steps summarize the calculation of the *predicted* B-plane vector from a Mars-centered position vector \mathbf{r} and velocity vector \mathbf{v} at the sphere-of-influence.

angular momentum vector

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

radius rate

$$\dot{r} = \frac{\mathbf{r} \cdot \mathbf{v}}{|\mathbf{r}|}$$

semiparameter

$$p = \frac{h^2}{\mu}$$

semimajor axis

$$a = \frac{r}{\left(2 - \frac{rv^2}{\mu}\right)}$$

orbital eccentricity

$$e = \sqrt{1 - p/a}$$

true anomaly

$$\cos \theta = \frac{p - r}{er}$$

$$\sin \theta = \frac{\dot{r}h}{e\mu}$$

B-plane magnitude

$$B = \sqrt{p|a|}$$

fundamental vectors

$$\hat{\mathbf{z}} = \frac{r\mathbf{v} - \dot{r}\mathbf{r}}{h}$$

$$\hat{\mathbf{p}} = \cos \theta \hat{\mathbf{r}} - \sin \theta \hat{\mathbf{z}}$$

$$\hat{\mathbf{q}} = \sin \theta \hat{\mathbf{r}} + \cos \theta \hat{\mathbf{z}}$$

S vector

$$\mathbf{S} = -\frac{a}{\sqrt{a^2 + b^2}} \hat{\mathbf{p}} + \frac{b}{\sqrt{a^2 + b^2}} \hat{\mathbf{q}}$$

B vector

$$\mathbf{B} = \frac{b^2}{\sqrt{a^2 + b^2}} \hat{\mathbf{p}} + \frac{ab}{\sqrt{a^2 + b^2}} \hat{\mathbf{q}}$$

T vector

$$\mathbf{T} = \frac{(S_y^2, -S_x^2, 0)^T}{\sqrt{S_x^2 + S_y^2}}$$

R vector

$$\mathbf{R} = \mathbf{S} \times \mathbf{T} = (-S_z T_y, S_z T_x, S_x T_y - S_y T_x)^T$$

Geocentric-to-areocentric coordinate transformation

This section describes the transformation of coordinates between the Earth mean equator and equinox of J2000 and areocentric mean equator and equinox of date coordinate systems. This transformation is used to compute the B-plane coordinates at encounter.

A unit vector in the direction of the pole of Mars can be determined from

$$\hat{\mathbf{p}}_{Mars} = \begin{bmatrix} \cos \alpha_p \cos \delta_p \\ \sin \alpha_p \cos \delta_p \\ \sin \delta_p \end{bmatrix}$$

The IAU 2000 right ascension and declination of the pole of Mars in the EME2000 coordinate system are given by the following expressions

$$\alpha_p = 317.68143 - 0.1061T$$

$$\delta_p = 52.88650 - 0.0609T$$

where T is the time in Julian centuries given by $T = (JD - 2451545.0)/36525$ and JD is the TDB Julian Date.

The unit vector in the direction of the *IAU-defined* x-axis is computed from

$$\hat{\mathbf{x}} = \hat{\mathbf{p}}_{J2000} \times \hat{\mathbf{p}}_{Mars}$$

where $\hat{\mathbf{p}}_{J2000} = [0\ 0\ 1]^T$ is unit vector in the direction of the pole of the J2000 coordinate system.

The unit vector in the y-axis direction of this coordinate system is

$$\hat{\mathbf{y}} = \hat{\mathbf{p}}_{Mars} \times \hat{\mathbf{x}}$$

Finally, the components of the matrix that transforms coordinates from the EME2000 system to the Mars-centered mean equator and equinox of date system are as follows:

$$\mathbf{M} = \begin{bmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{y}} \\ \hat{\mathbf{p}}_{Mars} \end{bmatrix}$$

References and Bibliography

“Update to Mars Coordinate Frame Definitions”, R. A. Mase, JPL IOM 312.B/015-99, 15 July 1999.

“JPL Planetary Ephemeris DE421”, E. M. Standish, JPL IOM 312.N-03-009, 24 April 2003.

“Report of the IAU/IAG Working Group on Cartographic Coordinates and Rotational Elements of the Planets and Satellites: 2000”, *Celestial Mechanics and Dynamical Astronomy*, **82**: 83-110, 2002.

“IERS Conventions (2003)”, IERS Technical Note 32, November 2003.

“Planetary Constants and Models”, R. Vaughan, JPL D-12947, December 1995.

“Preliminary Mars Planetary Constants and Models for Mars Sample Return”, D. Lyons, JPL IOM 312/99.DTL-1, 20 January 1999.

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

“Interplanetary Mission Analysis and Design”, Stephen Kemble, Praxis Publishing, 2006.

“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.

APPENDIX A

Compiling and Running the Software

This appendix describes how to compile and run the `itcm_soc`s computer program. This software was created using version 6.4.3 of SOCS and Compaq Visual Fortran.

A DOS/Windows version of `itcm_soc`s using Compaq Visual Fortran version 6.6C can be created with the following command:

```
f132 /arch:host itcm_soc.f *.for c:\socs\socs633.lib advapi32.lib
```

This command assumes the SOCS library is located in the subdirectory `c:\socs`.

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:

```
itcm_soc e2m1.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 itcm_soc           *
*                                         *
*           interplanetary TCM         *
*           optimization with SOCS     *
*                                         *
*           February 8, 2006           *
*****
```

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

The source code that reads the name of an input file included on the command line is

```
c   if present, use command line argument #1 for input file
    call getarg(1, inputfname$, istatus)
```

The source code that creates the file name input prompt is as follows:

```
c   clear screen

    isys = system("cls")

    if (istatus .eq. -1) then
c   *****
c   input filename not on command line
c   request name of simulation definition input file
c   *****
```

```

print *, ' '
print *, ' '

print *, ' *****'
print *, ' *          program itcm_socS          *'
print *, ' *          *          *'
print *, ' *          interplanetary TCM          *'
print *, ' *          optimization with SOCS        *'
print *, ' *          *          *'
print *, ' *          February 8, 2006            *'
print *, ' *****'
print *, ' '
print *, ' '

print *,
&      'please input the name of the simulation definition file'

      read (*, *) inputfname$
end if

```

If your compiler does not accept input from a command line, you will have to modify this source code for your particular Fortran compiler. You may also choose to eliminate the code that accepts a command line input file. Please note also that your compiler may have a different command to clear the screen.

Important Note

The binary ephemeris files provided with this computer program were created for use on PC-compatible computers. For other platforms, you will need to create 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/>.

APPENDIX B

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 `itcm_soc`s software.

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-vx` = x-component of the impulsive TCM velocity vector in meters/second
`delta-vy` = y-component of the impulsive TCM velocity vector in meters/second
`delta-vz` = z-component of the impulsive TCM velocity vector in meters/second
`deltav` = scalar magnitude of the impulsive TCM delta-v in meters/seconds

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
`rs2sc-x (au)` = x-component of spacecraft position vector in astronomical units
`rs2sc-y (au)` = y-component of spacecraft position vector in astronomical units
`rs2sc-z (au)` = z-component of spacecraft position vector in astronomical units
`rs2sc-mag (au)` = heliocentric radius magnitude of spacecraft in astronomical units
`vs2sc-x (km/sec)` = x-component of spacecraft velocity vector in kilometers per second
`vs2sc-y (km/sec)` = y-component of spacecraft velocity vector in kilometers per second
`vs2sc-z (km/sec)` = z-component of spacecraft velocity vector in kilometers per second

vs2sc-mag (km/sec) = heliocentric velocity of spacecraft in kilometers per second
rs2e-x (au) = x-component of Earth position vector in astronomical units
rs2e-y (au) = y-component of Earth position vector in astronomical units
rs2e-z (au) = z-component of Earth position vector in astronomical units
rs2e-mag (au) = heliocentric radius magnitude of Earth in astronomical units
rs2m-x (au) = x-component of Mars position vector in astronomical units
rs2m-y (au) = y-component of Mars position vector in astronomical units
rs2m-z (au) = z-component of Mars position vector in astronomical units
rs2m-mag (au) = heliocentric radius magnitude of Mars in astronomical units
sma (au) = transfer orbit semimajor axis in astronomical units
eccentricity = transfer orbit orbital eccentricity (non-dimensional)
inclination (deg) = transfer orbit orbital inclination in degrees
argper (deg) = transfer orbit argument of perigee in degrees
raan (deg) = transfer orbit right ascension of the ascending node in degrees
true anomaly (deg) = transfer orbit true anomaly in degrees

The heliocentric coordinates of the spacecraft are with respect to the Earth mean equator and equinox of J2000 coordinate system.. The areocentric coordinates of the spacecraft are with respect to the Mars-centered mean equator and equinox of date coordinate system.

APPENDIX C

Fortran Functions and Subroutines

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

- `itcm_soc`s.f - SOCS main executive program
- `atan3.for` - four quadrant inverse tangent function
- `eci2orb.for` - convert eci position and velocity vectors to classical orbital elements subroutine
- `findsoi.for` - subroutine that predicts time of entrance to sphere-of-influence
- `findca.for` - subroutine that predicts time of closest approach to Mars
- `fpaobj.for` - flight path angle objective function subroutine
- `gdate.for` - compute calendar date from Julian date subroutine
- `hel_eqm.for` - heliocentric equations of motion 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
- `odeigs.for` - SOCS initial guess subroutine
- `odeinp.for` - SOCS simulation input subroutine
- `odepf.for` - SOCS point functions subroutine
- `odeprt.for` - SOCS print subroutine - creates comma-separated-variable file
- `oderhs.for` - SOCS 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 position and velocity vectors in kilometers and kilometers/second
- `pci_eqm.for` - Mars-centered equations of motion subroutine

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
rmobj.for - Earth-centered distance 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 D

Example Fortran Subroutine

This appendix contains the source code for a single Fortran 77 routine and illustrates typical programming conventions used in the `itcm_soc`s software. This subroutine is the differential-algebraic routine required by the SOCS software.

```
      subroutine oderhs(iphase, t, ydyn, ny, p, np, frhs, nf, iferr)

c     computes the right hand sides of the
c     differential-algebraic (dae) equations

c     heliocentric equations of motion

c     output

c     frhs = right hand sides of differential-algebraic system

c     note: unit of position vector is astronomical units and
c           unit of components of velocity vector is au/day

c     *****

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

      include 'socscm1.inc'

      integer iphase, ny, np, nf, iferr

      dimension ydyn(ny), p(np), frhs(nf)

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

      dimension vplanet(3), rplanet(3), vtmp(3)

      dimension rp(9, 3), rp2sc(9, 3), rp2scm(9)

      dimension q(9), f(9), d3(9), asun(3), accp(3)

c     initialize evaluation error flag

      iferr = 0

c     current tdb julian date

      xjdate = xjdtdcm + t

c     -----
c     central body (sun) point mass gravity
c     -----

c     distance from sun to spacecraft

      rs2sc = sqrt(ydyn(1) * ydyn(1) + ydyn(2) * ydyn(2)
&             + ydyn(3) * ydyn(3))
```

```

do i = 1, 3
  asun(i) = -xmu(1) * ydyn(i) / rs2sc**3
end do

c -----
c planetary point mass perturbations
c -----

nplanets = 7

c calculate heliocentric position vectors of each planet
do i = 1, nplanets
  call sv2000(xjdate, i, 11, rplanet, vplanet)

  do j = 1, 3
    rp(i, j) = rplanet(j)
  end do
end do

c compute planetocentric position vectors of spacecraft
do i = 1, nplanets
  rp2sc(i, 1) = ydyn(1) - rp(i, 1)

  rp2sc(i, 2) = ydyn(2) - rp(i, 2)

  rp2sc(i, 3) = ydyn(3) - rp(i, 3)
end do

c compute planetocentric position magnitudes of spacecraft
do i = 1, nplanets
  rp2scm(i) = sqrt(rp2sc(i, 1)**2 + rp2sc(i, 2)**2
&               + rp2sc(i, 3)**2)
end do

c compute f(q) and other functions
do k = 1, nplanets

  do i = 1, 3
    vtmp(i) = ydyn(i) - 2.0d0 * rp(k, i)
  end do

  call vdot(ydyn, vtmp, dot1)

  dot2 = rp(k, 1) * rp(k, 1) + rp(k, 2) * rp(k, 2)
&       + rp(k, 3) * rp(k, 3)

  q(k) = dot1 / dot2

  f(k) = q(k) * ((3.0d0 + 3.0d0 * q(k) + q(k) * q(k))
&            / (1.0d0 + (1.0d0 + q(k))**1.5d0))

```

```

    d3(k) = rp2scm(k) * rp2scm(k) * rp2scm(k)
end do

do j = 1, 3
    accp(j) = 0.0d0

    do k = 1, nplanets
        accp(j) = accp(j) - xmu(k + 1)
&          * (ydyn(j) + f(k) * rp(k, j)) / d3(k)
    end do
end do

c    evaluate right hand sides of differential equations

frhs(1) = ydyn(4)

frhs(2) = ydyn(5)

frhs(3) = ydyn(6)

do i = 1, 3
    frhs(i + 3) = accp(i) + asun(i)
end do

return
end

```

APPENDIX E

Additional Program Examples

This appendix provides typical output from the `itcm_soc`s computer program for the two other types of user-defined final orbit targeting options.

The first example illustrates the solution for the “user-defined B-plane coordinates” program option.

```
-----
program itcm_soc
-----

input data file ==> e2m1.in

user-defined B-plane coordinates

TCM delta-v vector and magnitude
(heliocentric EME2000)
-----

delta-vx          2.739430078567175 meters/second
delta-vy          19.586485697202050 meters/second
delta-vz          -3.631140719029703 meters/second

deltav           20.107712004190198 meters/second

time and conditions prior to TCM maneuver
(heliocentric EME2000)
-----

calendar date      June      8, 2003
TDB time           18:20:44.077
TDB Julian date    2452799.264399034436792

      sma (km)          eccentricity      inclination (deg)      argper (deg)
0.190725765750D+09  0.204056802425D+00  0.234926769446D+02  0.253488459798D+03

      raan (deg)        true anomaly (deg)      arglat (deg)          period (min)
0.463121032996D+00  0.394861259377D+01  0.257437072392D+03  0.757158581013D+06

      rx (km)           ry (km)                 rz (km)               rmag (km)
-.319331575699D+08  -.136207676243D+09  -.590899587841D+08  0.151867971768D+09

      vx (kps)          vy (kps)                vz (kps)              vmag (kps)
0.316260608115D+02  -.655290820823D+01  -.295930905686D+01  0.324330978868D+02

time and conditions after TCM maneuver
(heliocentric EME2000)
-----

calendar date      June      8, 2003
TDB time           18:20:44.077
```

TDB Julian date 2452799.264399034436792

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.190708902912D+09	0.203958859298D+00	0.234969610289D+02	0.253617319269D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.511255263319D+00	0.377560940567D+01	0.257392928675D+03	0.757058168045D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.319331575699D+08	-.136207676243D+09	-.590899587841D+08	0.151867971768D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.316288002416D+02	-.653332172254D+01	-.296294019758D+01	0.324321493593D+02

time and conditions at Mars SOI
(heliocentric EME2000)

calendar date December 23, 2003

TDB time 11:05:45.489

TDB Julian date 2452996.962332044262439

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.187247982388D+09	0.200438520384D+00	0.234408534753D+02	0.252506549732D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.353516246510D+00	0.153359677528D+03	0.458662272599D+02	0.736543698162D+06
rx (km)	ry (km)	rz (km)	rmag (km)
0.151572164373D+09	0.145114747432D+09	0.625129579930D+08	0.218952690899D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.144059014072D+02	0.157697959383D+02	0.687596287887D+01	0.224387014954D+02

time and conditions at Mars SOI
(Mars-centered mean equator and equinox of date)

calendar date December 23, 2003

TDB time 11:05:45.489

TDB Julian date 2452996.962332044262439

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.584625823159D+04	0.185494233945D+01	0.600085894333D+02	0.113970437610D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105683072723D+03	0.240802911511D+03	0.354773349120D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.338026864652D+05	0.145661656775D+06	-.118347159607D+05	0.150000001229D+06
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.549131216934D+00	-.273187812332D+01	0.363479696844D+00	0.281012822912D+01

b-plane coordinates at Mars SOI
(Mars-centered mean equator and equinox of date)

b-magnitude 9133.664978411341508 kilometers
b dot r -7887.885860399128433
b dot t 4604.898760143742038
theta 300.276117249008962 degrees
vinf 2.706617012552135 km/sec
r-periapsis 4998.213689545828856 kilometers
decl-asy 7.486317261314183 degrees
rasc-asy 281.333407587247621 degrees

flight path angle -86.637777897223600 degrees

time and conditions at Mars closest approach
(Mars-centered mean equator and equinox of date)

calendar date December 23, 2003

TDB time 00:00:41.283

TDB Julian date 2452996.500477812718600

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.585170397067D+04	0.185413935843D+01	0.600037956316D+02	0.113964057134D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105656643368D+03	0.951621960694D-11	0.113964057134D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.165082037875D+04	-.257097731307D+04	0.395557553733D+04	0.499817067523D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.218650777486D+01	-.408039240408D+01	-.173958622237D+01	0.494535930402D+01

b-plane coordinates at Mars closest approach
(Mars-centered mean equator and equinox of date)

b-magnitude 9136.593482882577518 kilometers
b dot r -7889.908529926355186
b dot t 4607.242544174102477
theta 300.282415492444159 degrees
vinf 2.705357297352535 km/sec
r-periapsis 4998.170675234768169 kilometers
decl-asy 7.505180208407690 degrees
rasc-asy 281.295026508628723 degrees

flight path angle 6.174947147579107E-012 degrees

time and conditions of Mars at closest approach
(heliocentric EME2000)

calendar date December 23, 2003

TDB time 00:00:41.283

TDB Julian date 2452996.500477812718600

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.152368095650D+01	0.935424561181D-01	0.246772247412D+02	0.332979394152D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.337165765323D+01	0.698323876969D+02	0.428117818485D+02	0.540647074387D-06
rx (km)	ry (km)	rz (km)	rmag (km)
0.152344907282D+09	0.144377764077D+09	0.621052753985D+08	0.218885757342D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.164717428205D+02	0.170512689472D+02	0.826604265201D+01	0.251075993551D+02

transfer time 197.236078778281808 days

This next example illustrates the solution for the “grazing flyby; user-defined B-plane angle” program option.

program itcm_soc

input data file ==> e2m1.in

grazing flyby; user-defined b-plane angle

TCM delta-v vector and magnitude
(heliocentric EME2000)

delta-vx 2.940658902297949 meters/second
delta-vy 20.007420714662047 meters/second
delta-vz -2.833100853033079 meters/second
deltav 20.419863341281420 meters/second

time and conditions prior to TCM maneuver
(heliocentric EME2000)

calendar date June 8, 2003

TDB time 18:20:44.077

TDB Julian date 2452799.264399034436792

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.190725765750D+09	0.204056802425D+00	0.234926769446D+02	0.253488459798D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.463121032996D+00	0.394861259377D+01	0.257437072392D+03	0.757158581013D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.319331575699D+08	-.136207676243D+09	-.590899587841D+08	0.151867971768D+09

vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.316260608115D+02	-.655290820823D+01	-.295930905686D+01	0.324330978868D+02

time and conditions after TCM maneuver
(heliocentric EME2000)

calendar date	June 8, 2003
TDB time	18:20:44.077
TDB Julian date	2452799.264399034436792

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.190709588262D+09	0.203960536184D+00	0.234967432664D+02	0.253627215182D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.508812972245D+00	0.376795327428D+01	0.257395168456D+03	0.757062249007D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.319331575699D+08	-.136207676243D+09	-.590899587841D+08	0.151867971768D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.316290014704D+02	-.653290078752D+01	-.296214215771D+01	0.324321879138D+02

time and conditions at Mars SOI
(heliocentric EME2000)

calendar date	December 23, 2003
TDB time	12:30:45.258
TDB Julian date	2452997.021357151679695

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.187245731077D+09	0.200450376977D+00	0.234417327515D+02	0.252510115658D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.353761262271D+00	0.153389412887D+03	0.458995285456D+02	0.736530414839D+06
rx (km)	ry (km)	rz (km)	rmag (km)
0.151487676619D+09	0.145202804620D+09	0.625537112185D+08	0.218964237775D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.144166127983D+02	0.157595023042D+02	0.687184393443D+01	0.224370870772D+02

time and conditions at Mars SOI
(Mars-centered mean equator and equinox of date)

calendar date	December 23, 2003
TDB time	12:30:45.258
TDB Julian date	2452997.021357151679695

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.584744452334D+04	0.158077970300D+01	0.602921459752D+02	0.120630164156D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105606088605D+03	0.233441992059D+03	0.354072156215D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.327433867654D+05	0.145762912908D+06	-.134552550745D+05	0.150000000151D+06
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.547805129594D+00	-.273188698078D+01	0.363369682848D+00	0.280986378001D+01

b-plane coordinates at Mars SOI
(Mars-centered mean equator and equinox of date)

```

-----
b-magnitude          7158.916428383650782 kilometers
b dot r              -6200.534029741367704
b dot t              3578.192584895416985
theta                299.988304315629819 degrees
vinf                 2.706342448142359 km/sec
r-periapsis         3396.077093560152207 kilometers
decl-asy             7.473093939703507 degrees
rasc-asy            281.313794494247077 degrees

flight path angle    -87.365311874756699 degrees

```

time and conditions at Mars closest approach
(Mars-centered mean equator and equinox of date)

```

-----
calendar date        December 22, 2003
TDB time             23:55:37.518
TDB Julian date      2452996.496962011326104

```

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.586082146036D+04	0.157947337020D+01	0.602844121020D+02	0.120609354373D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.105539842949D+03	0.360000000000D+03	0.120609354373D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.932643532227D+03	-.205423547395D+04	0.253858207061D+04	0.339618996375D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.270199564719D+01	-.434359448904D+01	-.252218243936D+01	0.570341983551D+01

b-plane coordinates at Mars closest approach
(Mars-centered mean equator and equinox of date)

```

-----
b-magnitude          7165.405244358685195 kilometers
b dot r              -6205.422979279391257
b dot t              3582.702606150030533
theta                299.999999851998552 degrees
vinf                 2.703252160046220 km/sec
r-periapsis         3396.189963748778155 kilometers
decl-asy             7.524067849209975 degrees

```

rasc-asy 281.216518858666973 degrees

flight path angle -1.721771298254188E-013 degrees

time and conditions of Mars at closest approach
(heliocentric EME2000)

calendar date December 22, 2003

TDB time 23:55:37.518

TDB Julian date 2452996.496962011326104

sma (au)	eccentricity	inclination (deg)	argper (deg)
0.152368095823D+01	0.935424573343D-01	0.246772247408D+02	0.332979394488D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.337165765228D+01	0.698303981358D+02	0.428097926235D+02	0.540647075306D-06
rx (km)	ry (km)	rz (km)	rmag (km)
0.152349910736D+09	0.144372584410D+09	0.621027644258D+08	0.218885110914D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.164711571685D+02	0.170518239673D+02	0.826628139803D+01	0.251076706851D+02

transfer time 197.232562976889312 days