Closest Approach Between the Earth and Heliocentric Objects

This ***Numerit*** application (`cae2ho`) uses a combination of Cowell's method and one-dimensional minimization to predict closest approach conditions between the Earth and objects such as asteroids, comets and other bodies in heliocentric elliptic orbits. It includes the point mass gravity perturbations due to the planets Mercury, Venus, Earth, Mars, Jupiter and Saturn. It also includes the point-mass effect of the Moon via a combined Earth/Moon gravitational constant. The coordinates of the planets are determined with the SLP96 ephemeris. The software expects to find this ephemeris in a binary data file named `slp96.bin`.

The second-order heliocentric equations of motion of a satellite or celestial body subject to the point mass gravitational attraction of the Sun and planets are given by

$$\frac{d^2\boldsymbol{r}}{dt^2}(\boldsymbol{r},t) = -m_s \frac{\boldsymbol{r}_{s-b}}{\left|\boldsymbol{r}_{s-b}\right|^3} - \sum_{i=1}^{9} m_{p_i}\left(\frac{\boldsymbol{r}_{(p-b)_i}}{\left|\boldsymbol{r}_{(p-b)_i}\right|^3} + \frac{\boldsymbol{r}_{p_i}}{\left|\boldsymbol{r}_{p_i}\right|^3}\right) \qquad (1)$$

where

$m_s =$ gravitational constant of the Sun
$m_{p_i} =$ gravitational constant of planet i
$\boldsymbol{r}_{p_i} =$ position vector from the Sun to planet i
$\boldsymbol{r}_{s-b} =$ position vector from the Sun to the body
$\boldsymbol{r}_{(p-b)_i} =$ position vector from planet i to the body

These position vectors are related according to

$$\boldsymbol{r}_{s-b} = \boldsymbol{r}_p + \boldsymbol{r}_{p-b} \qquad (2)$$

This ***Numerit*** program uses a Runge-Kutta-Fehlberg 7(8) method to numerically integrate the first-order form of these orbital equations of motion. This is a variable step size method of order 7 with an 8th order error estimate which is used to dynamically change the integration step size during the simulation.

The syntax of this ***Numerit*** function is as follows:

```
function rkf78 (neq, ti, tf, h, tetol, x, xout)

` solve first order system of differential equations

` Runge-Kutta-Fehlberg 7(8) method

` input

`  neq   = number of differential equations
`  ti    = initial simulation time
```

```
`  tf    = final simulation time
`  h     = initial guess for integration step size
`  tetol = truncation error tolerance (non-dimensional)
`  x     = integration vector at time = ti

` output

`  xout  = integration vector at time = tf
```

This function requires that the following statement be executed prior to the first time `rkf78` is called.

```
rkcoef = 1
```

This statement is placed in the main program and is used to initialize the integration coefficients for this numerical method. The value of `rkcoef` is passed to the `rkf78` function using a `common rkcoef` statement in the main program. The `rkf78` function will then set this "flag" to 0 after the coefficients are calculated.

This function also requires a second function which defines the user's system of first-order vector differential equations. The format of this function is given by

```
function orbeqm (time, y, ydot)

` first order equations of orbital motion

` cowell's method

` input

`  time = simulation time (seconds)
`  y    = state vector

` output

`  ydot = integration vector
```

where `orbeqm` is the actual name of the user-coded function. The function parameter list should be exactly as shown here. The main program uses "rerouting" to tell the `rkf78` function which equations of motion function to use. For this example the code is

```
orbeqm -> caeqm
```

This software also uses a one-dimensional optimization algorithm due to Richard Brent to solve the close approach problem. Additional information about this numerical method can be found in *Algorithms for Minimization Without Derivatives*, R.P. Brent, Prentice-Hall, 1972. As the title of this book indicates, this algorithm does not require derivatives of the objective function. This feature is important because the analytic first derivative of many objective functions is difficult to derive and code. The objective function for this classical celestial mechanics problem is the scalar geocentric distance of the heliocentric body.

The software will ask you for the name of a simple ASCII input file which defines the simulation. When responding to this request be sure to include the file name extension. The software will also ask you for the total simulation (search) duration in days and an Earth-to-body close approach constraint in astronomical units. The software will ignore any close approaches that are larger than the close approach constraint. Please note the proper units and coordinate system (J2000 equinox) of the input orbital elements. Notice also that time on the Barycentric Dynamical Time (TDB) scale is used.

The following is a typical input data file (`xf11n.dat`) for this program. It contains the J2000 dynamical equator and equinox heliocentric orbital elements of the asteroid 1997 XF 11. This file was created using data available on the Horizons ephemeris system which is located on the Internet at http://ssd.jpl.nasa.gov/horizons.html. Additional information about NEOs (Near Earth Objects) can also be found at http://newton.dm.unipi.it/neodys/.

Do not change the total number of lines or the order of annotation and data in this file. The software expects to find exactly 34 lines of information in the input data file. The last data item of this file defines the fundamental plane of the J2000 coordinate system.

```
initial calendar date
(1 <= month <= 12, 1 <= day <= 31, year = all digits!)
10,20,1998

initial TDB
(0 <= hours <= 24, 0 <= minutes <= 60, 0 <= seconds <= 60)
0,0,0

initial semimajor axis (AU)
(semimajor axis > 0)
0.1441779254846407D+01

initial orbital eccentricity (non-dimensional)
(0 <= eccentricity < 1)
0.4836740409964595D+00

initial orbital inclination
(degrees; 0 <= inclination <= 180)
0.2017287833484568D+02

initial argument of perihelion
(degrees; 0 <= argument of perihelion <= 360)
0.3228014264370198D+03

initial right ascension of the ascending node
(degrees; 0 <= raan <= 360)
0.3533298230482567D+03

initial mean anomaly
(degrees; 0 <= mean anomaly <= 360)
0.2708752703925974D+03

coordinate frame (1 = ecliptic, 2 = equator)
2
```

The conversion of a position $\boldsymbol{r}$ and velocity vector $\boldsymbol{v}$ from the J2000 ecliptic frame (subscript *ec*) to the J2000 equatorial frame (subscript *eq*) is given by the following matrix-vector relationship:

$$\begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \end{bmatrix}_{eq} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varepsilon & -\sin\varepsilon \\ 0 & \sin\varepsilon & \cos\varepsilon \end{bmatrix} \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \end{bmatrix}_{ec} \tag{3}$$

In this expression ε is the J2000 obliquity of the ecliptic.

The following is the draft output created by this program for this example. The program output and all internal calculations are performed in the J2000 dynamical equinox and equator coordinate system. All angular elements in this output are in degrees.

```
initial heliocentric orbital elements
(J2000 dynamical equinox and equator)

October 20, 1998

00 h 00 m 00 s

semimajor axis (au)                1.441779254846407
eccentricity                       0.4836740409964595
inclination                        20.17287833484568
argument of perihelion             322.8014264370198
right ascension of ascending node  353.3298230482567
true anomaly                       221.6544086526512


close approach conditions

October 31, 2002

00 h 30 m 40.5942 s

heliocentric orbital elements
(J2000 dynamical equinox and equator)

semimajor axis (AU)                1.442267124200949
eccentricity                       0.4841223848226988
inclination                        20.16564547782856
argument of perihelion             322.7924266589294
right ascension of ascending node  353.3467738153024
true anomaly                       77.11093827535139

geocentric distance (AU)           0.06355264422584024
```