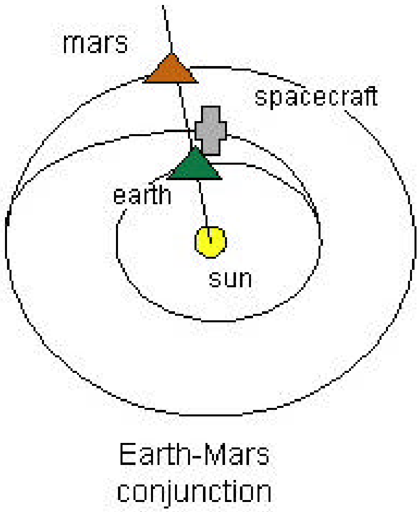
The Two Body Problem

& Applications



by
Wm. H. Clark II

1. Newton's Law of Gravity

Classical Orbital Mechanics begins and is thereafter based upon just one equation, Newton's Universal Law of Gravitation.

(1) $$F = -G\frac{m_1 m_2}{r^2}$$

Which, in vector form is

(2) $$\vec{F} = -G\frac{m_1 m_2}{r^3}\vec{r}$$

Newton also defined Force as

(3) $$\vec{F} = m\vec{a} = m\ddot{\vec{r}}$$

These important relations can be used to study the motion of two bodies in an inertial reference frame.

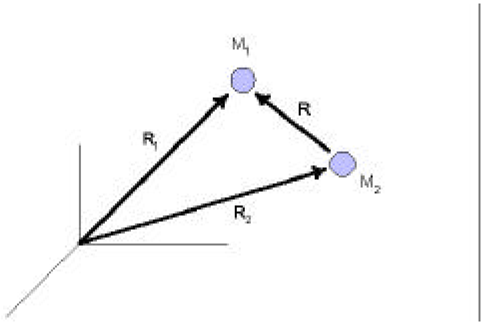


Figure 1 Two bodies in a coordinate system

Thus,

(4) $$\vec{r} = \vec{r}_1 - \vec{r}_2$$

From (2), the force on the two bodies, respectfully, is

(6a) $$\cancel{m_1}\ddot{\vec{r}}_1 = -G\frac{\cancel{m_1}m_2}{r}(\vec{r}_1-\vec{r}_2) = +G\frac{m_1 m_2}{r^3}\vec{r}$$

(6b) $$\cancel{m_2}\ddot{\vec{r}}_2 = -G\frac{m_1\cancel{m_2}}{r}(\vec{r}_2-\vec{r}_1) = -G\frac{m_1 m_2}{r^3}\vec{r}$$

Substituting these two equations into (5)

$$\ddot{\vec{r}} = -G\frac{m_1+m_2}{r^3}(\vec{r}_1-\vec{r}_2) = G\frac{m_1+m_2}{r^3}\vec{r}$$

And the result is the important <u>Two Body Equation</u>

(7) $$\ddot{\vec{r}} = -\frac{\mu}{\mathrm{r}^3}\vec{r}$$

Where μ is the <u>Gravitational Constant</u> for the system of two bodies

$$\mu = G(m_1+m_2)$$

Often $m_2 << m_1$, e.g. an artificial satellite orbiting Earth, in which case

$$\mu \cong Gm_1$$

Another important relation is obtained by cross multiplying the radius vector into (7) the Two Body Equation

$$\vec{r}\times\left[\ddot{\vec{r}} = -\frac{\mu}{\mathrm{r}^3}\vec{r}\right]$$

from which $\vec{r}\times\vec{r}=0$, which leaves $\vec{r}\times\ddot{\vec{r}}=0$. Now observe that

$$\vec{r}\times\ddot{\vec{r}} = \frac{d}{dt}\left(\vec{r}\times\dot{\vec{r}}\right)$$

Thus,

$$\vec{r} \times \ddot{\vec{r}} = constant$$

This constant of proportionality is a vector and by convention it is called the Angular Momentum vector

(9) $$\vec{r} \times \dot{\vec{r}} = \vec{h}$$

This important vector is perpendicular to the plane of the orbit, of the small body m_2 about the much larger body m_1.

Another important constant is obtained by taking the dot product of the velocity and (7) the Two Body Equation.

$$\dot{\vec{r}} \bullet \left[\ddot{\vec{r}} = -\frac{\mu}{r^3} \vec{r} \right]$$

$$\dot{\vec{r}} \bullet \ddot{\vec{r}} = \dot{\vec{r}} \bullet \frac{\mu}{r^3} \vec{r} = r\dot{r} \frac{\mu}{r^3} \quad \textit{(a scalar)}$$

Observe now that

$$\frac{d}{dt}\left(\frac{\dot{\vec{r}}^2}{2} \right) = \dot{\vec{r}} \bullet \ddot{\vec{r}} = \dot{r}\ddot{r} \quad \textit{(a scalar)}$$

$$\frac{d}{dt}\left(\frac{\mu}{\vec{r}} \right) = -\frac{\mu}{\vec{r}^2} \dot{\vec{r}} = -\frac{\mu}{\vec{r}^3} \vec{r}\dot{\vec{r}} = -\frac{\mu}{r^3} r\dot{r} = -\frac{\mu}{r^2} \dot{r}$$

Thus,

$$\frac{\dot{r}^2}{2} - \frac{\mu}{r} = constant$$

This constant of proportionality is a scalar and it is called by convention Energy, because the terms are analogous to kinetic energy and potential energy in analytical mechanics.

(10) $$\frac{v^2}{2} - \frac{\mu}{r} = E$$

The analysis so far has assumed an inertial reference frame. This notion can be developed formally by first assuming the absence of any external forces, so

$$\sum \vec{F} = m_1\ddot{\vec{r}}_1 + m_2\ddot{\vec{r}}_2 = 0$$

(that is, add equations (6)) then integrate twice to get

$$m_1\dot{\vec{r}}_1 + m_2\dot{\vec{r}}_2 = \vec{a}$$

(11) $$m_1\vec{r}_1 + m_2\vec{r}_2 = \vec{a} + \vec{b}t$$

This is the equation of motion for the Center of Mass – it is constant, as required of an inertial reference system. That is, the reference frame is not accelerating but moving at a constant velocity.

From this brief study, ten Constants of Motion have been validated: six from the two vectors in (11), three from the angular momentum vector (9) and one from the scalar energy integral (10). No other constants are known to exist for the Two Body Problem.

Discussion

A rule implicit in Celestial Mechanics is that a geometric proof is the only real, irrefutable evidence for Laws of nature. Newton derived the Universal Law of Gravitation without the use of the calculus, using only principles of geometry. Read Principia and discuss his work in concept, if not in detail.

2. Kepler's First Law

Kepler's First Law states that the orbit of each planet is an ellipse with the sun at one focus. Isolating on the sun and one planet – the Two Body Problem – consider the geometry of the ellipse.

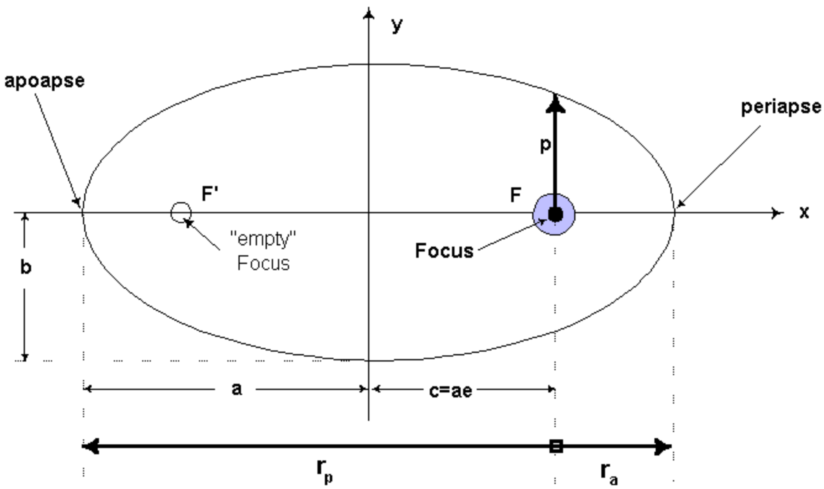


Figure 1 The ellipse

The following equations define the dimensions of an ellipse:

$$a^2 = b^2 + c^2 \qquad b = \sqrt{1-e^2}$$

$$e = \frac{c}{a} = \frac{r_a - r_p}{r_a + r_p} \qquad p = \frac{b^2}{a} = a\left(1-e^2\right)$$

$$r_p = a(1-e) \qquad r_a = a(1+e)$$

where: a = semi-major axis, b = semi-minor axis
e = eccentricity, p = semi-parameter
r_p = radius at periapse, r_a = radius at apoapse

From analytic geometry the equation of an ellipse is

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

Where the two foci are at (±c,0) and the vertices are at (±a,0) in the (x,y) coordinate system of Figure 1, which has the origin midway between the foci. A more convenient reference system in the study of orbits places the origin at the primary focus (c,0) where the central body is situated. The angle of the spacecraft (s/c) on the ellipse is measured with respect to the positive x-axis which goes through the periapse. This angle is called the true anomaly, f.

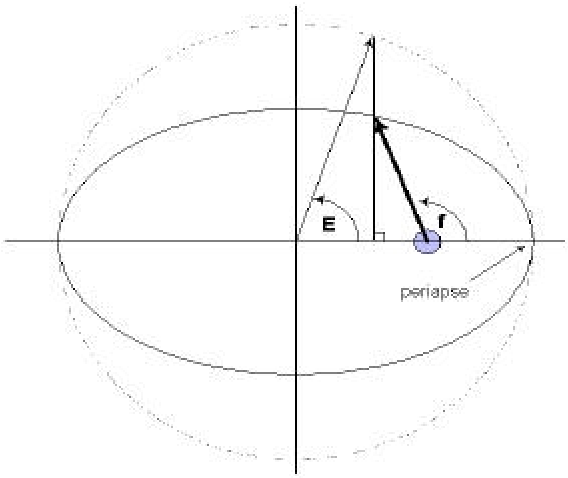


Figure 2 Eccentric Anomaly

Another important angle is the eccentric anomaly, E which is measured from the origin of a circle of radius a to the line intersecting this circle at the extension of f. The eccentric anomaly is constructed in Figure 2.

The position of the s/c on the ellipse in terms of E is

(1a) $x = a(\cos E - e)$

(1b) $y = b(\sin E)$

The radius vector to the s/c from $r^2 = x^2 + y^2$ is

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

after substituting equations (1) for x and y, then simplifying. An important relationship between the true and eccentric anomaly is

(3) $$\tan\left(\frac{f}{2}\right) = \sqrt{\frac{1+e}{1-e}}\tan\left(\frac{E}{2}\right)$$

An expression for the radius in terms of the true anomaly can be found from the geometry of the hyperbola.

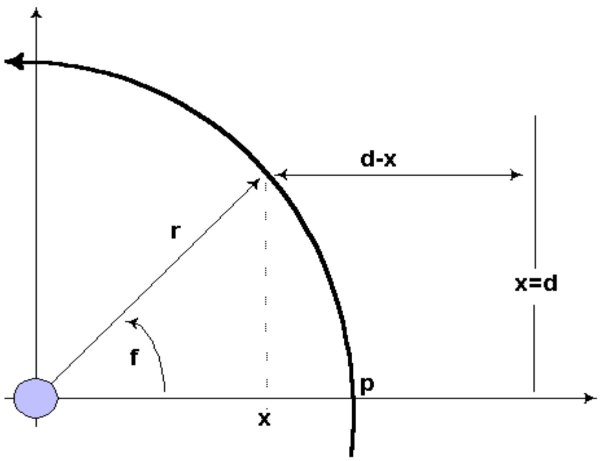


Figure 3 Focus of the hyperbola

The hyperbola is defined by its eccentricity, which is the ratio of the distance of any point on the hyperbola from the focus versus the distance from a vertical line x=d.

(4) $$e = \frac{r}{d - x}$$

The semi-parameter is defined by

(5) $$p = ed$$

From Figure 3, it is also known that

(6) $$x = r \cos f$$

From equation (4),

$$r = ed - ex = p - e\left(r \cos f\right)$$

Solving for r, the result is known as the trajectory equation.

(6) $$r = \frac{p}{1 + e \cos f}$$

This equation is valid for all conic sections: the ellipse, parabola, and hyperbola.

Discussion

The eccentric anomaly has a geometric significance, if not a practical real world one. The mean anomaly has no such geometric analogy, except that for orbits of small eccentricity it is approximately the angle (constructed the same way as the eccentric anomaly) from the empty focus. Find the maximum eccentricity for which this approximation is valid. Construct a set of equations using this approximation and propagate an orbit without iterating Kepler's equation. Compare the speed and accuracy of this algorithm with methods that iterate to get the eccentric anomaly.

3. Time in Elliptical Motion

An important vector constant was discovered by Laplace called the eccentricity vector.

(1) $$\vec{L} = \dot{\vec{r}} \times \vec{h} - \mu \frac{\vec{r}}{r} = \dot{\vec{r}} \times \vec{h} - \mu \hat{r}$$

where $\hat{r}$ is a vector of unit length. Taking the time derivative of $\vec{L}$

$$\dot{\vec{L}} = \ddot{\vec{r}} \times \vec{h} - \mu \dot{\hat{r}}$$

$$= -\mu \frac{\vec{r}}{r^3} \times \vec{h} - \mu \frac{\vec{h} \times \vec{r}}{r^3} = 0$$

Thus $\vec{L}$ is constant in magnitude. Furthermore it is clear that $\vec{L} \times \vec{h} = 0$, so the two vectors are not independent. From the algebra,

$$\vec{L} \bullet \vec{r} = h - \mu r = Lr \cos\theta$$

where θ is the angle between $\vec{L}$ and $\vec{r}$. Solving for r,

(2) $$r = \frac{\frac{h^2}{\mu}}{1 + \left(\frac{L}{\mu}\right)\cos\theta} = \frac{p}{1 + e\cos f}$$

Comparing this to equation (2.7), it is easy to see that θ is the true anomaly. Thus, $\vec{L}$ points toward periapse and

(3) $$\vec{e} = \frac{\vec{L}}{\mu} \quad \text{and so,} \quad e = \frac{L}{\mu}$$

Thus the name “eccentricity vector,” since the length of $\vec{L}$ is the magnitude of the eccentricity of the ellipse. Another important relation is for the semi-parameter

(4) $$p = \frac{h^2}{\mu} \quad \text{or} \quad h = \sqrt{\mu p}$$

It is now possible to consider the last “orbital element” ~ time, the rate at which the s/c orbits the central body. From analytic geometry,

(4) $$dA = ½ r^2 df$$

because $A = \pi r^2$ and $A_{\text{sector}} = ½(\theta) r^2$ This is now illustrated.

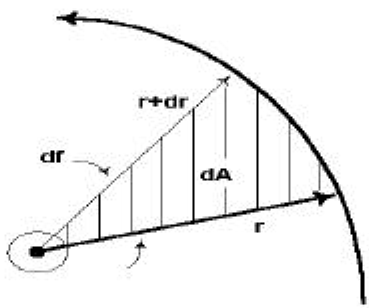


Figure 1 Geometry of Kepler’s 2nd Law

From basic dynamic principles,

$$v_\theta = r\omega = r\dot{\theta} = r\dot{f} = r\frac{df}{dt}$$

where

$$v_\theta = v\sin\theta = v\sin f$$

And since

$$h = rv\cos f = rv_\theta \Rightarrow v_\theta = \frac{h}{r}$$

Solving for dt among these equations,

(6) $$dt = \frac{2}{h} dA$$

Which is a formal statement of Kepler's 2nd Law. That is, h is constant so equal areas are swept out in equal times by the radius vector from the focus to the orbiting s/c.

The area of an ellipse is πab so the period of an orbit can be found by integrating (6) over 2pi radians. First defining the period of the orbit,

(7) $$\tau = \frac{2}{h}\left(\pi ab\right)$$

then substituting (4) for h,

(8) $$\tau = 2\pi\sqrt{\frac{a^3}{\mu}} = 2\pi n$$

which is a statement of Kepler's Third Law. The value of n, the mean motion is

(9) $$\mu = n^2 a^3$$

where this motion defines a new angle called the mean anomaly.

(10) $$M = n\left(t - t_p\right)$$

where t_p is the time of periapse passage by the orbiting s/c.

An analytic expression for the mean anomaly can be found from the geometry of the ellipse versus the eccentric anomaly.

The derivation is most easily understood in a geometric context.

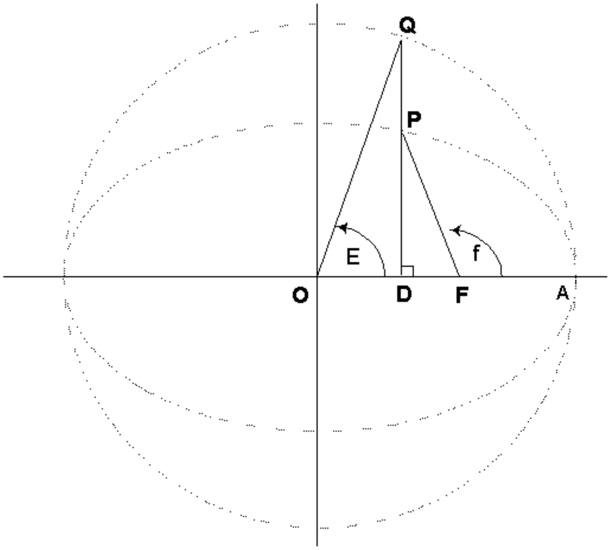


Figure 2 Geometry of the eccentric anomaly

$$\frac{M}{2\pi} = \frac{area \sim AFP}{area \sim ellipse} = \frac{area \sim AFQ}{area \sim circle}$$

$$AFQ = ACQ - FCQ$$

$$= ½a^2 E - ½ae\left(a \sin E\right)$$

$$\frac{M}{2\pi} = \frac{a^2\left(E - e\sin E\right)}{2\pi a^2}$$

Which reduces to Kepler’s Equation.

(11) $$M = E - e\sin E$$

Discussion

About 99.5% of all satellites in orbit are in nearly circular orbits. So it's understandable that nobody has ever noticed that equation (4) in this famous and fundamental derivation is for circles and not ellipses. Not even Newton noticed, when he derived Kepler's 2nd law analytically. That is, he studied the area between two radius vectors in the limit; saying that as they get closer together they approach equal length. Alas, two different radius vectors of equal length mean the orbit is a circle. Create a set of test ellipses of varying eccentricity and determine how large an error is introduced in the orbital elements. Kepler's 3rd law is derived from the 2nd ~ estimate the magnitude of the errors in that formulation; and then in angular momentum.

4. Orbital Elements

The orbit can be described in a right handed coordinate system called IJK coordinates. The vector **I** is fixed in inertial space, directed to the First Point or Aries; **J** is at an angle of ninety degrees such that the **IJ** plane is the Earth's equatorial plane; and **K** is toward the North pole.

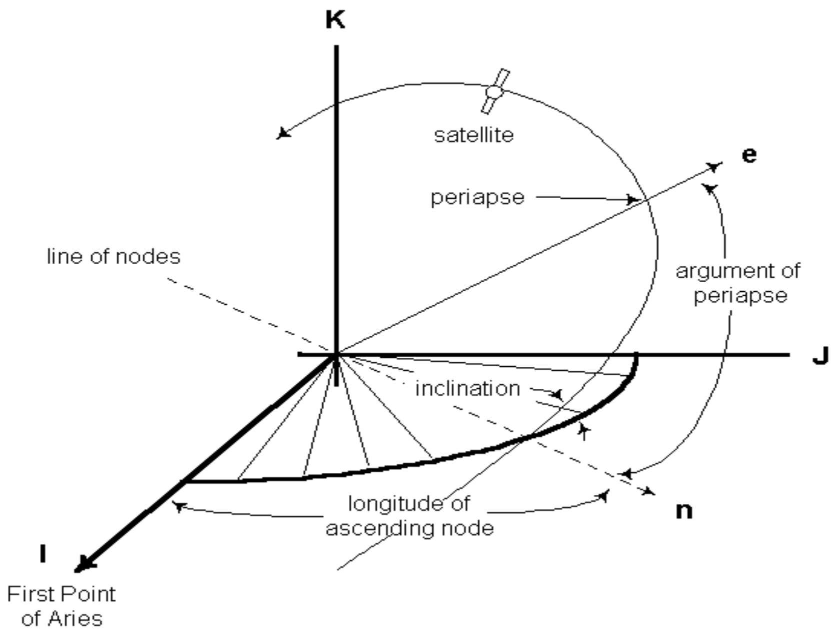


Figure 1 A satellite orbit in the IJK coordinate system

The orbital plane and the **IJ** plane intersect along the vector **n,** the line of nodes; with the orbital plane rotated about this vector by an amount equal to the inclination of the orbit. The longitude of the ascending node of the orbit is measured in the Earth's equatorial plane as shown, and the argument of periapse is the angle measured in the plane of the orbit to the vector **e** through periapse. The symbols for these angles are

i = inclination of the orbit
ω = argument of the periapse
Ω = longitude of the ascending node

Position and velocity in the plane of the orbit can be calculated directly from the orbital elements.

(1) $$\vec{r} = \frac{p}{1 + e\cos f}\begin{bmatrix} \cos f \\ \sin f \\ 0 \end{bmatrix}$$

(2) $$\vec{v} = \sqrt{\frac{\mu}{p}}\begin{bmatrix} -\sin f \\ \cos f + e \\ 0 \end{bmatrix}$$

Given $\vec{r}$ and $\vec{v}$ in the plane of the orbit, the orientation in the **IJK** system can be determined by the following series of calculations

(3) $$\vec{h} = \vec{r} \times \vec{v}$$

(4) $$\vec{e} = \frac{1}{\mu}\left(\vec{v} \times \vec{h}\right) - \frac{\vec{r}}{r}$$ where $e = \left|\vec{e}\right|$

(5) $$\vec{n} = \vec{k} \times \vec{h}$$ where $\vec{k} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$

(6) $$\cos i = \frac{\vec{k} \bullet \vec{h}}{kh}$$ or $\cos i = \frac{h_z}{h}$ or $\sin i = n$

(7) $$\cos\Omega = \frac{\vec{I} \bullet \vec{n}}{In}$$ and if $r_x < 0$ then $\Omega = 360 - \Omega$

(8) $$\cos\omega = \frac{\vec{n} \bullet \vec{e}}{ne}$$ and if $e_z < 0$ then $\omega = 360 - \omega$

(9) $$\cos f = \frac{\vec{e} \bullet \vec{r}}{er}$$ and if $\vec{r} \bullet \vec{v} < 0$ then $f = 360 - f$

Translating coordinates from **PQW** to **IJK** is done by a series of three rotations about the respective axes. These rotations must be done in the order specified from right to left.

(10) $$\vec{r}_{IJK} = ROT3(-\Omega)\,ROT1(-i)\,ROT3(-\omega)\,\vec{r}_{PQW}$$

where the rotations about the x- y- and z-axes in a right handed coordinate system are represented by the following matrices

(11) $$ROT1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos & \sin \\ 0 & -\sin & \cos \end{bmatrix}$$

(12) $$ROT2 = \begin{bmatrix} \cos & 0 & -\sin \\ 0 & 1 & 0 \\ \sin & 0 & \cos \end{bmatrix}$$

(13) $$ROT3 = \begin{bmatrix} \cos & \sin & 0 \\ -\sin & \cos & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

It is interesting to notice that

(14) $$\vec{r}_{IJK} = \begin{bmatrix} \hat{P} & \hat{Q} & \vec{W} \end{bmatrix} \vec{r}_{PQW}$$

where the column vectors of the single 3x3 transformation matrix from **PQW** to **IJK**in equation (10) are unit vectors in the orbital plane. That is, the vector **P** configured in the orbital plane points in the direction of periapse (i.e. the eccentricity vector); the vector **Q** in the orbital plane is at a true anomaly of ninety degrees towards the semi-perimeter; and the vector **W** in the orbital plane is perpendicular to the plane of the orbit, directed upwards. These three vectors form the axes of the PQW coordinate system, which is the formal name of the right hand oriented inertial reference frame used in identifying the orbital elements of a, e, p, n, f, M, and E in previous sections.

Latitude and longitude are found by rotating the **IJK** coordinates through another angle, the angle equivalent of Greenwich Standard Time (GST) of the site

(15) $$\vec{r}_{ECEF} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} = ROT3(\theta_{GST})\vec{r}_{IJK}$$

(16a) longitude, $\lambda = \arctan\left(\frac{r_y}{r_x}\right)$

(16b) latitude, $\phi = \arctan\left(\frac{r_z}{r}\right)$

Then, azimuth and elevation are calculated from the range vector, as follows.

(17) $$\vec{\rho}_{IJK} = \vec{r}_{IJK} - \vec{r}_{site,IJK}$$

(18) $$\vec{\rho}_{SEZ} = ROT2(90° - \phi)ROT3(\lambda)ROT3(\theta_{GST})\vec{\rho}_{IJK}$$

$$= \rho \begin{bmatrix} -\cos(el)\cos(Az) \\ \cos(el)\sin(Az) \\ \sin(el) \end{bmatrix} = \begin{bmatrix} \rho_x \\ \rho_y \\ \rho_z \end{bmatrix}$$

Where,

(19a) $$Az = \arctan\left(-\frac{\rho_y}{\rho_x}\right)$$

(19b) $$El = \arcsin\left(\frac{\rho_z}{\rho}\right)$$

5. Orbit Propogation

Given the time when the spacecraft (s/c) is at periapse, its position in the orbit at some later time can be determined. There are two suitable methods and both start by solving Kepler's Equation.

The orbital elements can be determined from the initial position and velocity. Then the mean anomaly is found from

$$M = n(t - t_p)$$

The eccentric anomaly must now be found from Kepler's Equation

$$M = E - e\sin E$$

A common way to solve this transcendental equation for E is Newton-Raphson Iteration. Rearranging the function,

$$E - e\sin eE - M = 0$$

it can be expanded in a Taylor Series about $f(y) = 0$ for $y = x + \delta$

$$0 = f(x) + f'(x)\,\delta + ...$$

where

(1) $$\delta = -\frac{f(x)}{f'(x)} = \frac{E - e\sin E - M}{1 - e\cos E}$$

Typically, algorithms start with an initial guess of $E_1 = M$ and then

(2) $$E_{n+1} = E_n + \frac{M - E_n + e\sin E_n}{1 - e\cos E_n}$$

A better first guess for E comes from the series expansion

$$E = M + e\sin M + \frac{1}{2}e^2 \sin 2M + \frac{1}{8}e^3\left(3\sin 3M - \sin M\right)$$

A second method of orbit propagation, using the f and g functions, works for all conic sections. It applies a principle from linear algebra that if three vectors are coplanar ($\vec{r}, \vec{r}_0, \dot{\vec{r}}, \dot{\vec{r}}_0$ are all in the plane of the orbit) then one vector can be expressed as the sum of the two other vectors, if they are not collinear. Thus

(3) $\vec{r} = f\vec{r}_0 + g\dot{\vec{r}}_0$

differentiating once,

(4) $\dot{\vec{r}} = \dot{f}\vec{r}_0 + \dot{g}\dot{\vec{r}}_0$

A useful relation (i.e. the determinate of equations (3) and (4) as rows) is

(5) $\dot{g}f - g\dot{f} = 1$

Which shows that $f, g, \dot{f}, \dot{g}$ are not independent. However, if any three are known then equation (5) can be used to solve for the fourth variable. The functions in terms of the orbital elements are as follows

(6a) $f = 1 - \frac{a}{r_0}\left(1 - \cos\theta\right)$ where $\theta = E - E_0$

(6b) $g = \tau - \frac{\theta - \sin\theta}{n}$ where $\tau = t - t_0$

(6c) $\dot{f} = -\frac{na^2}{rr_0}\sin\theta$

(6d) $\dot{g} = 1 - \frac{a}{r}\left(1 - \cos\theta\right)$

A variation of equations (3) and (4) can be used to find an earlier position (e.g. periapse if it's not known) from an existing position and velocity vector by

(7a) $$\vec{r}_0 = \dot{g}\vec{r} - g\dot{\vec{r}}$$

(7b) $$\dot{\vec{r}}_0 = -\dot{f}\vec{r} + f\dot{\vec{r}}$$

These two equations come from taking the inverse of the f and g functions,

$$\begin{bmatrix} f & g \\ \dot{f} & \dot{g} \end{bmatrix}^{-1} = \begin{bmatrix} \dot{g} & -g \\ -\dot{f} & f \end{bmatrix} \frac{1}{\Delta} \text{ where } \Delta = 1 \text{ from (5)}$$

Another interesting property of the f and g functions is that they are themselves (scalar) solutions to the two body equation, perhaps implying they have some fundamental significance.

(8a) $$\ddot{f} = -\frac{\mu}{r^3} f$$

(8b) $$\ddot{g} = -\frac{\mu}{r^3} g$$

The leading terms of series expansions in time of the f and g functions (i.e. those terms that are known) are useful when the initial position vector is known but not the initial velocity vector.

(9a) $$f = 1 - \frac{1}{2}\sigma\left(t - t_0\right)^2 + \frac{1}{2}\sigma\psi\left(t - t_0\right)^3 + ...$$

(9b) $$g = \left(t - t_0\right) - \frac{1}{6}\sigma\left(t - t_0\right)^3 + ...$$

where $\sigma = \frac{\mu}{r_0^3}$ and $\psi = \frac{\dot{r}_0}{r_0}$.

The f and g functions can be used to study the individual components of the position and velocity vectors.

(10a) $$x = fx_0 + g\dot{x}_0$$

(10b) $$y = fy_0 + g\dot{y}_0$$

(11a) $$\dot{x} = \dot{f}x_0 + \dot{g}\dot{x}_0$$

(11b) $$\dot{y} = \dot{f}y_0 + \dot{g}\dot{y}_0$$

where,

(12) $$\det\begin{bmatrix} x_0 & \dot{x}_0 \\ y_0 & \dot{y}_0 \end{bmatrix} = x_0\dot{x}_0 - \dot{x}_0 y_0 = h$$

From which the magnitude of the angular momentum vector can be used to determine the initial Flight Path Angle from

$$h = r\dot{r}\cos\phi$$

Discussion

Establish a set of criteria by which to judge the two methods of orbit propagation presented here – the direct method and using f and g functions. Grade each method on speed, accuracy, elegance, and overall computational efficacy. Identify specific situations for which each method is better suited.

6. Initial Orbit Determination

An important angle in orbit determination is the flight path angle. This is the angle between the local horizontal and the velocity vector of a spacecraft on a conic section trajectory.

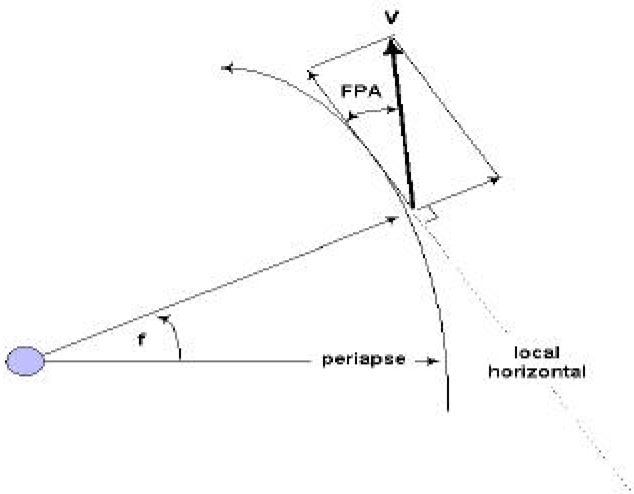


Figure 1 Flight path angle

The velocity vector can be written in terms of the following scalar quantities

(1a) $\dot{r} = v \sin \beta_{FPA}$

(1b) $r\dot{f} = v \cos \beta_{FPA}$

where $\vec{h} = \vec{r} \times \vec{v} = rv \cos\left(\beta_{FPA}\right) = rv\left(\dfrac{r\dot{f}}{v}\right)$

which can be solved for the following useful scalar equation

(2) $h = r^2 \dot{f}$

Consider the problem of initial orbit determination. Upon insertion into a new orbit after launch, the range (magnitude of the position vector), speed (magnitude of the velocity vector) and flight path angle are determined by surface radar ranging. The orbital elements can be found:

$$h = r_0 v_0 \cos \beta_0$$

$$E = \frac{v_0^{\;2}}{2} - \frac{\mu}{r_0} = -\frac{\mu}{2a} \Rightarrow \text{solve for a}$$

$$e^2 = 1 + \frac{2Eh^2}{\mu^2} \Rightarrow \text{ solve for e}$$

$$p = a\left(1 - e^2\right) \quad \text{(check: } h^2 = \mu p \text{)}$$

$$r = \frac{p}{1 + e\cos f} \Rightarrow \text{ solve for } f$$

When determining the flight path angle from orbital elements, it is important to have find both cosine and sine to resolve quadrant issues using the atan2 command. Two useful formulas are

(5a) $$\sin \beta = \frac{e \sin f}{\sqrt{1 + 2e\cos f + e^2}}$$

(5b) $$\cos \beta = \frac{1 + e\cos f}{\sqrt{1 + 2e\cos f + e^2}}$$

Initial orbit determination is not often this easy. In practical situations for insertion into Earth orbits (in contrast to similar situations in interplanetary missions, to be considered later; where the initial range, speed, and flight path angle are in fact often known), the information available is limited to radar ranging which does not provide either initial velocity or flight path angles. Some intricate calculations must be done, which will be outlined here but not described in complete detail.

The situation is as follows: **R** is known (the observer's position on Earth), and you are trying to find **r** (the position of the s/c in Earth centered coordinates). Ground radar finds three ranges and angles, which are used to approximate the true range. The vector equation of the situation is:

(6) $$\vec{\rho} = \vec{R} + \vec{r}$$

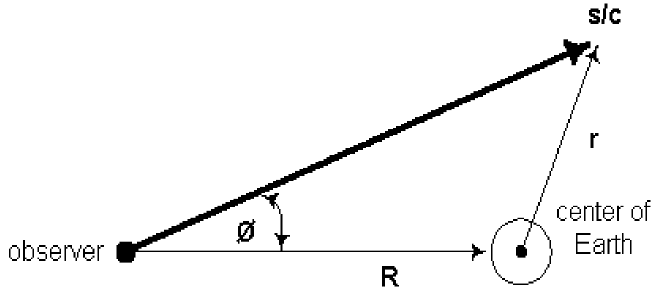


Figure 2 The unknown range vector geometry

The Laplace method of initial orbit determination can be used when the position and velocity vectors must be determined from three range vectors of the s/c soon after it is placed in orbit.

The Laplace method is geometric in nature. It interpolates the known vectors into a suitable fit. A low order polynomial is used to approximate the orbital path (e.g. forcing a fit of the polynomial to the three observations), then approximate values are found for the unit vectors toward the object and the derivatives of this unit vector.

The Laplace method proceeds as follows:

(i) find the (x,y,z) components of the three range vectors $\hat{\rho}_{-1}, \hat{\rho}_0, \hat{\rho}_1$

(ii) use differencing functions to approximate the $\dot{\hat{\rho}}$ and $\ddot{\hat{\rho}}$

(iii) use the cosine law to solve for the unknowns

(7) $$r^2 = \rho^2 + R^2 - 2\rho R \cos\phi$$

The Gauss method starts with the same information as the Laplace method – three initial ranges (with angles of inclination) but solves for the state vector by approximating the dynamics of the orbit. Where the state vector is a six element column vector of the position and velocity vectors at any given time:

(8) $$\vec{X} = \begin{bmatrix} \vec{r} \\ \dot{\vec{r}} \end{bmatrix} = \begin{bmatrix} r_x \\ r_y \\ r_z \\ \dot{r}_x \\ \dot{r}_y \\ \dot{r}_z \end{bmatrix}$$

The dynamics is approximated using a truncated series in time; that is, by using the f and g series.

(9a) $$\vec{r}_{-1} = f_{-1}\vec{r}_0 + g_{-1}\dot{\vec{r}}_0$$

(9b) $$\vec{r}_1 = f_1\vec{r}_0 + g_1\dot{\vec{r}}_0$$

An expression for ρ_0 is derived, and then the following equation is solved iteratively for the unknowns using the cosine law

(10) $$\rho_0 = \left(A + \frac{B}{r_0^3} \right)$$

It is then possible to solve for ρ_{-1} and ρ_1 , and then to find $\dot{\vec{r}}_0$ using differencing and the f and g series.

Discussion

The f and g series are similar to power series. Their application in solving the initial orbit determination problem implies they serve – as is common with power series – to generate a solution to a problem near a singularity. In this case the singularity is caused by the very short time between observed ranges; meaning the denominator in the Two Body Equation is going to zero. The issue is that a single power series would not work. Discuss possible reasons why a bicameral series is required. (You will need to depend upon geometric arguments such as in the Laplace method; just as Newton derived the Law of Gravity using geometric reasoning.)

7. Perturbations

The analysis so far has been of an idealized Two Body Problem. Several assumptions that were made to develop the equations will continue to be valid as more practical situations are studied.

(i) the mass of the s/c is negligible in relation to the central body

(ii) motion takes place in an inertial reference frame

The second criteria is valid for typical Earth satellites, to the accuracy considered here, but it becomes an important issue for interplanetary missions which will be considered later.

Two other assumptions made so far will not be negligible in practical mission planning for near Earth missions.

(iii) the two bodies are spherically symmetric point masses

(iv) no forces act on the bodies other than gravitational forces

It is still appropriate to study orbital motion as a Two Body Problem, but some perturbations to the conic motion(s) will be added.

<u>Solar radiation pressure</u> effects are usually small for a satellite unless it has low mass and large surface area. The force generated is

(1) $$m_{sat}\ddot{\vec{r}} = -P_{SR}C_R A_\odot \hat{r}$$

where $\hat{r}$ = unit vector from the s/c to the Sun

$$P_{SR} = \frac{1353 W/m^2}{c}$$ where c = speed of light in m/sec

$A_\odot$ = are exposed (perpendicular) to the Sun

C_R = 0 if transparent (reflectivity of $A_\odot$)
= 1 if black surface
= 2 if reflective surface

The second perturbing force to consider is <u>atmospheric drag</u>. It is a far more ubiquitous phenomenon – especially for near Earth satellites – and it can perturb the motion drastically. The magnitude of the drag force is

(2) $$m_{sat}\ddot{\vec{r}} = -\tfrac{1}{2} C_D A \rho v_{REL}^2 \hat{v}$$

where C_D = <u>coefficient of drag</u>

A = cross sectional area perpendicular to $\hat{v}$

$\dfrac{C_D A}{m}$ = <u>ballistic coefficient</u> for a specific s/c orientation

v_{REL} = with respect to the rotating atmosphere ($\vec{\omega}_{\oplus} \times \vec{r}$)

$\hat{v}$ = unit vector of the s/c velocity vector

Drag has the most prominent influence upon the semi-major axis

(3) $$\dot{a} = \frac{da}{dt} = -\left(\frac{C_D A}{m}\right)\frac{\rho a^2}{\mu} v_{IJK}^2$$

Drag tends to circularize orbits by decreasing a and decreasing e in such a way that the periapse is constant and the apoapse decreases.

Satellites above low Earth orbits can be changed more by <u>third body perturbations</u> – e.g. from the moon or sun – than by atmospheric drag. The force is strictly due to gravity but the scale and units can cause numerical problems in computer models. These are avoided by using the equation

(4) $$\ddot{\vec{r}}_{\oplus sat} = -\frac{G m_{\oplus} \vec{r}_{\oplus sat}}{r_{\oplus sat}^3} + G m_3 \left[\frac{\vec{r}_{sat3}}{r_{sat3}^3} - \frac{\vec{r}_{\oplus 3}}{r_{\oplus 3}^3}\right]$$

where the $\oplus$ symbol is for Earth.

The three terms of this equation are called, in order of their appearance, the two body term, the direct term and the indirect term.

By far the most important perturbation for low Earth satellites is that caused by the equatorial bulge of the Earth called the J2 effect (after the term in the series expansion of the Earth's gravitational field). The influence on the orbital elements are

(5) $$\dot{\Omega} = -\frac{3}{2}[\ \ ]\cos i$$

(6) $$\dot{\omega} = \frac{3}{4}[\ \ ]\left(5\cos^2 i - 1\right)$$

(7) $$\dot{M} = \frac{3}{4}[\ \ ]\left(3\cos^2 i - 1\right)\sqrt{1-e^2}$$

(8) $$\dot{a} = \dot{e} = \frac{di}{dt} = 0$$

where $$[\ \ ] = \frac{nR_{\oplus}^2 J2}{p^2}$$

The relative magnitudes of these perturbations for a typical satellite are as follows

Force	with respect to the central body
2-body	1.0
J2	1.0E-4
Sun	0.5E-4
Moon	0.5E-6
Drag	1.0E-8
Solar Rad.	0.5E-9

Discussion

Discuss how "perturbations" or errors in the fundamental equations might introduce what seem to be physical perturbations. Pose experiments that would, for example, distinguish between such conceptual errors in the physical model and say relativistic errors.

8. Perturbed Orbit Determination

Finding the expected position of existing satellites in Earth orbit at some future time is a simple matter of downloading the s/c two line element set (TLE) from the internet, and performing the calculations developed thus far for the two body problem, before applying some simple perturbation formulas. The first thing is to extract the data from the TLE, each entry having two rows of data identified by an ID# as follows; the important information is

$$
\begin{array}{cccccccc}
\# & \sim & epoch & \sim & \frac{\dot{n}}{2} & \frac{\ddot{n}}{6} & \sim \\
\# & i & \Omega & .e & \omega & M & \bar{n}
\end{array} \tag{1}
$$

where epoch = time of periapse passage

$\dot{n}/2$ = used in perturbation calculations

$\ddot{n}/6$ = used in Taylor Series expansion of perturbations

i, Ω, ω = given in degrees (convert to radians)

$.e$ = eccentricity (leading decimal point assumed)

M = mean anomaly (iterate with e to for eccentric anomaly)

$\bar{n}$ = mean motion in revs/day (convert to n, rads/sec)

These TLE values can be used to determine a full set of orbital elements for the orbit of the s/c at the epoch time. First be sure all the units are consistent, then iterate to get E, from M and e. Then:

$$\frac{f}{2} = \arctan\left[\sqrt{\frac{1+e}{1-e}}\tan\left(\frac{E}{2}\right)\right]$$

$$a^2 = \frac{\mu}{n^2} \Rightarrow a \text{ and } p = a(1-e^2)$$

Perturbations cause this orbit at epoch to change over time (the TLE is accurate for about two weeks after the epoch time). New orbital elements at some later time can be determined as follows:

$$i = i_0$$

$$n_0 = \sqrt{\frac{\mu}{a_0^3}}$$

$$p_0 = a_0\left(1 - e_0\right)^3$$

$$a = a_0 - \frac{2}{3}\frac{a_0}{n_0}\dot{n}\Delta t$$

$$e = e_0 - \frac{2}{3}\frac{(1 - e_0)}{n_0}\dot{n}\Delta t$$

$$M = M_0 + n_0\Delta t + \frac{\dot{n}}{2!}\Delta t^2 + \frac{\ddot{n}}{3!}\Delta t^3$$

$$\Omega = \Omega_0 + \Omega_{J2}$$

$$\omega = \omega_0 + \omega_{J2}$$

Where the terms $\frac{\dot{n}}{2!}$ and $\frac{\ddot{n}}{3!}$ for M come from the TLE.

A more rigorous method – i.e. the one used to calculate the TLE itself – applies sophisticated statistical methods. The approach is as follows:

(i) Obtain preliminary values for the position and velocity vectors using the Laplace or Gauss method.

(ii) Compare this to a best case model (e.g. a Two Body model) of the orbit, and calculate a <u>reference solution</u>

(2) reference solution = observed – calculated values

The state vector $\vec{X}$ for the problem includes not only the usual position and velocity coordinates but also estimated values for quantities used in the model – e.g. the value for J2 or the gravitational constant.

(3)
$$\vec{X} = \begin{bmatrix} r_x \\ r_y \\ r_z \\ \dot{r}_x \\ \dot{r}_y \\ \dot{r}_z \\ \mu \\ J2 \\ etc \\ \dots \end{bmatrix} \text{; the reference solution = } \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \dot{x} \\ \Delta \dot{y} \\ \Delta \dot{z} \\ \Delta J2 \\ \Delta \mu \\ etc \\ \dots \end{bmatrix} = \begin{bmatrix} \Delta X_1 \\ \Delta X_2 \\ \Delta X_3 \\ \Delta X_4 \\ \Delta X_5 \\ \Delta X_6 \\ \Delta X_7 \\ \Delta X_8 \\ etc \\ \dots \end{bmatrix}$$

(iii) Form a state transition matrix which is how each variable changes with respect to all others:

(4)
$$\dot{\Phi} = \begin{bmatrix} \frac{\partial X_1}{\partial x} & \frac{\partial X_1}{\partial y} & \frac{\partial X_1}{\partial z} & \dots \\ \frac{\partial X_2}{\partial x} & \frac{\partial X_2}{\partial y} & \dots & \\ \frac{\partial X_3}{\partial x} & \dots & & \\ \dots & & & \end{bmatrix}$$

These variational equations – e.g. individual elements in the state transition matrix – can be found analytically (forming partial derivatives by hand) for small problems; using computer generated partials such as created by mathematica for larger problems; or numerically when the derivatives are too complex for even mathematica.

(iv) The state transition matrix is calculated using the best known values. The initial values are the position and velocity found by the Gauss or Laplace methods, plus best estimates for J2, the gravitational constant, and any other variables to be estimated in the problem.

(v) A matrix problem is solved using standard methods of linear algebra.

(5)
$$\begin{bmatrix} \Delta X_1 \\ \Delta X_2 \\ \Delta X_3 \\ \cdots \end{bmatrix} = \begin{bmatrix} \frac{\partial X_1}{\partial x} & \frac{\partial X_1}{\partial y} & \cdots & \\ \frac{\partial X_2}{\partial x} & & & \\ \cdots & & & \\ & & & \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \cdots \end{bmatrix}$$

known (obs. – ref.)   known (calculated)   solve for these

(vi) Form a new reference solution using

(6)
$$\Delta X_1' = \Delta X_1 + \Delta x$$
$$\Delta X_2' = \Delta X_2 + \Delta y$$

(vii) Solve for new corrections to the solution

(viii) Go to (vi) until the solution converges to the desired resolution.

Discussion

Exact solutions are known for a whole family of Three Body Problems – the circular problem, various elliptical problems, all involving two major bodies and a small third body. The TLE is based upon the simple Two Body model and its conventional set of orbital elements. The 3BP model is far more accurate, and it has many variations – circular, elliptical, etc. Write a proposal for a research project to study the possibility of using better fundamental models for the TLE so that the solution will be more accurate, and for a longer period of time.

9. Orbit Transfer

The most efficient way to move a s/c to another orbit is using a <u>Hohmann Transfer</u>, which takes 180º of longitude to make the transition along an elliptical path.

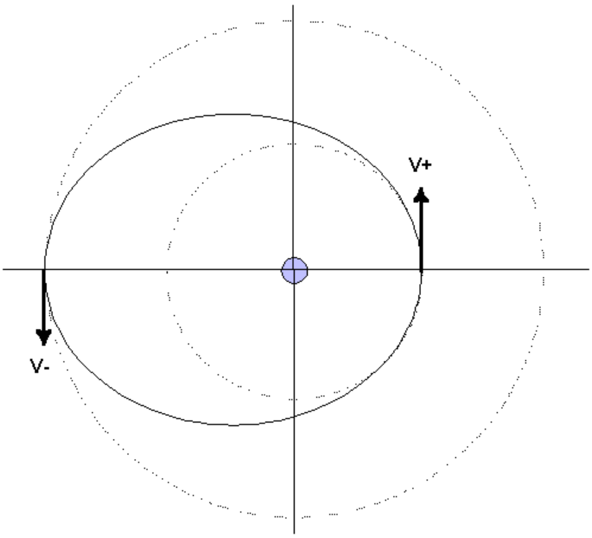


Figure 1 Hohmann Transfer

The orbit transfer requires two tangential thrusts (i.e. co-linear to the velocity vector) at periapse and apoapse of the transfer ellipse. The first (v+) increases the circular velocity to put the s/c on an elliptical trajectory and the second (v-) increases the elliptical velocity to put the s/c on the final circular path. Key orbital elements of the transfer ellipse are

(1a) $$a_{trans} = \frac{r_1 + r_2}{2}$$

(1b) $$e_{trans} = \frac{r_1 - r_2}{r_1 + r_2}$$

(1c) $$h = r_p v_p = r_a v_a$$

The simplest case is when the initial and final orbits are circular, in which case the circular velocities are

(2a) $$v_{initial} = \sqrt{\frac{\mu}{r_1}}$$

(2b) $$v_{final} = \sqrt{\frac{\mu}{r_2}}$$

If the initial and/or final orbit is an ellipse instead of a circle, the initial and final velocities are the respective orbital velocities on each of these orbits.

The velocities on the transfer ellipse can be found using the vis viva equation.

(3) $$v^2 = \mu\left(\frac{2}{r} - \frac{1}{a}\right)$$

Thus,

(4a) $$v^+ = \sqrt{\frac{2\mu}{r_1} - \frac{\mu}{a_{trans}}}$$

(4b) $$v^- = \sqrt{\frac{2\mu}{r_2} - \frac{\mu}{a_{trans}}}$$

From which the initial and final thrusts are

(5a) $$\Delta v_1 = v^+ - v_{initial}$$

(5b) $$\Delta v_2 = v^- - v_{final}$$

If the initial and/or final orbit is not circular the thrusts will not be tangential to the velocity. The thrust must then be calculated using the cosine law.

If there is an object – e.g. a satellite, space station, or planet – in the target orbit, then the orbital maneuver is called a rendezvous.

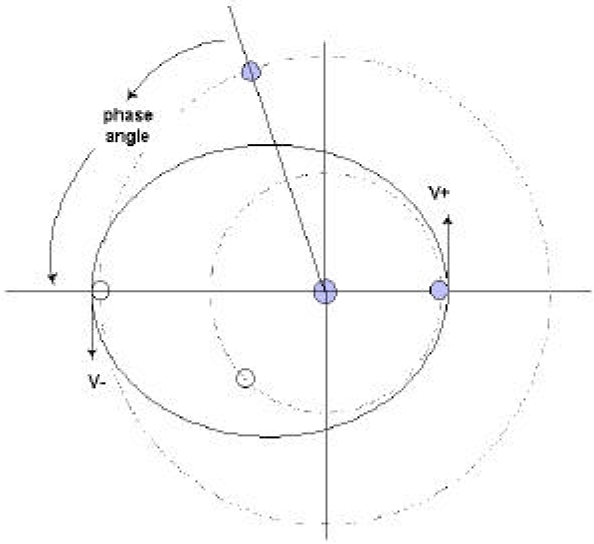


Figure 2 Rendezvous

The first thrust must occur so that the time of flight on the transfer ellipse is the time it takes the target to move through the phase angle.

(6) $\Delta t = \pi\sqrt{\frac{a_{trans}^3}{\mu}}$ and $\Delta\theta = n_{target}\Delta t$

(if the rendezvous is to a lower orbit be sure $r_p > r_\oplus$)

Given an angular separation between the two objects, the time for the optimum angular separation is determined by

(7) $\frac{\Delta\theta}{\tau}$ where $\tau = n_{intercept} - n_{target}$

The synodic period, τ , is the time needed to repeat a specific configuration of two bodies in orbit around a common central body.

If there is a great disparity between the sizes of the two orbits the target may make one or more complete revolutions while the intercept s/c is in transit. An algorithm to study this problem follows

(a) $$n_{tgt} = \sqrt{\frac{\mu}{a^3}}$$

(b) $$\tau_{phase} = \frac{k_{t\arg et}(2\pi)+\theta}{n_{t\arg et}}$$ where k=1,2,3,...

(c) $$a_{transfer} = \left[\mu\left(\frac{\tau_{phase}}{2\pi k_{initial}}\right)^2\right]^{1/3}$$

(d) $$r_p = 2a - r_a$$

(e) IF $r_p > 6378$; else $k_{t\arg et}$=2; etc... GOTO (b)

(f) $$\Delta v_{total} = 2\left|\sqrt{\frac{2\mu}{a_{t\arg et}} - \frac{\mu}{a_{phase}}} - \sqrt{\frac{\mu}{a_{t\arg et}}}\right|$$

Discussion

Actually, a <u>bi-elliptic</u> orbit transfer is more efficient than a Hohmann Transfer, but it takes longer. Take an Earth to Mars mission and model it using two elliptical transfers, Earth to conjunction and conjunction to Mars. Apply a thrust and conjunction, in addition to the two Hohmann trusts. Show that it is possible to reach Mars faster with the same total thrust as a Hohmann transfer; conversely that reaching Mars in the same time as a Hohmann requires less total thrust.

10. Lambert's Problem

A common problem in mission planning is to find a flight path that goes between two known position vectors. The analysis gets its name from a theory proposed by Lambert that the transit time is a function of only the semi-major axis of the solution ellipse, the sum of the magnitudes of the position vectors, and the chord length.

There are many applications of this problem, e.g. when a spacecraft is to take other than a minimum energy Hohmann Transfer between to trajectories. It might be a rendezvous or the other spacecraft might be targeted for impact. There is a whole family of curves that include any given two radius vectors, but there is among these just one optimal path. It is also the simplest trajectory to find; that is, the minimum energy path between two position vectors.

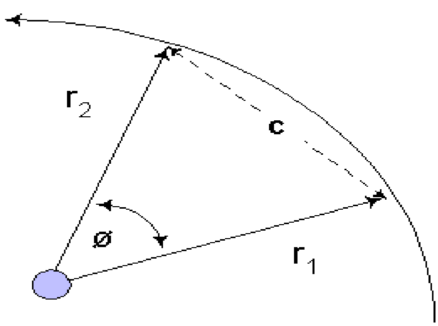


Figure 1 Lambert's Problem

Several features are in evidence from the geometry and from the cosine law:

(1) $$\cos\phi = \frac{\vec{r}_1 - \vec{r}_2}{r_1 r_2} \quad \text{where } \phi = \Delta f$$

(2) $$c^2 = r_1^2 + r_2^2 - 2 r_1 r_2 \cos\phi$$

Considering the two vectors in context of the minimum energy ellipse, it is possible to find the semi-major axis from the geometry of the problem.

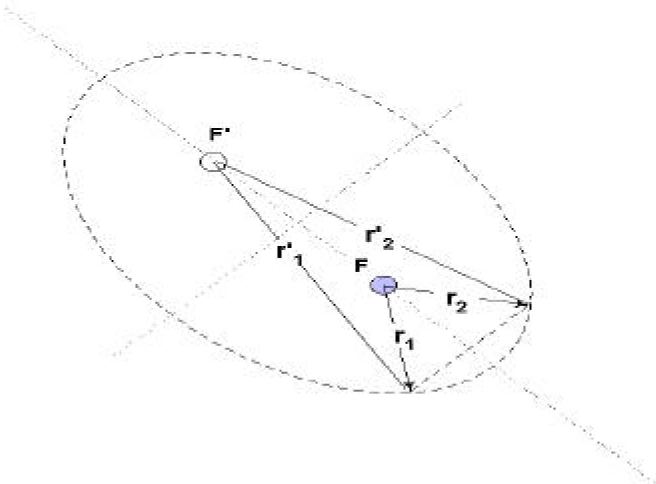


Figure 2 The minimum energy solution

The minimum energy transfer path is when the semi-major axis is the shortest.

(3) $$E = -\frac{\mu}{2a}$$

The shortest semi-major axis is when the two vectors from the primary focus are co-linear; that is, if they coincide with the chord. By the geometry, the major axis is always equal to the length of the line from one focus to a point on the ellipse, then to the other focus

(4a) $$r_1^{'} + r_1 = 2a$$

(4b) $$r_2^{'} + r_2 = 2a$$

Adding equations (4) and solving for the minimum a,

(5) $$a_{\min} = \frac{r_1 + r_2 + c}{4}$$

Another elegant formula exists for the eccentricity of this ellipse

(6) $$e_{\min} = \frac{|r_2 - r_1|}{2}$$

The velocity vector at the first position vector is

(7) $$\vec{v}_1 = \frac{h}{r_1 r_2 \sin\phi}\left\{\vec{r}_2 - \left[1 - \frac{r_2}{p}\left\langle 1 - \cos\phi \right\rangle\right]\vec{r}_1\right\}$$

The time of flight is found using the two Lambert angles

(8) $$\cos\alpha = 1 - \frac{r_1 - r_2 + c}{2a}$$

(9) $$\cos\beta = 1 - \frac{r_1 + r_2 - c}{2a}$$

These two angles appear in the Lambert equation for time of flight.

(10) $$\Delta t = \left(\frac{1}{n}\right)\left[\left(\alpha - \sin\alpha\right) - \left(\beta - \sin\beta\right)\right]$$

This equation confirms Lambert's theorem. Although the semi-parameter can be calculated from the formulas for a and e, it can also be found by

(11) $$p = \frac{2}{c}\left(s - r_1\right)\left(s - r_2\right)$$

where s is the semi-perimeter

(12) $$s = \frac{r_1 + r_2 + c}{2}$$

which, by inspection, is equal to 2a ~ the major axis of the minimum energy solution.

Discussion

Find a minimum energy Lambert solution to a Hohmann transfer. Do this for a full Hohmann, then for the second half of a bi-elliptic transfer from Earth to Mars; and also for the second half of the Mars to Earth mission. Theoretically, having more degrees of freedom should make it possible to do better than the Hohmann transfer.

11. Ballistic Trajectories

An important problem in orbital mechanics is the ballistic trajectory, a scenario where there are no thrusts applied after the initial burn.

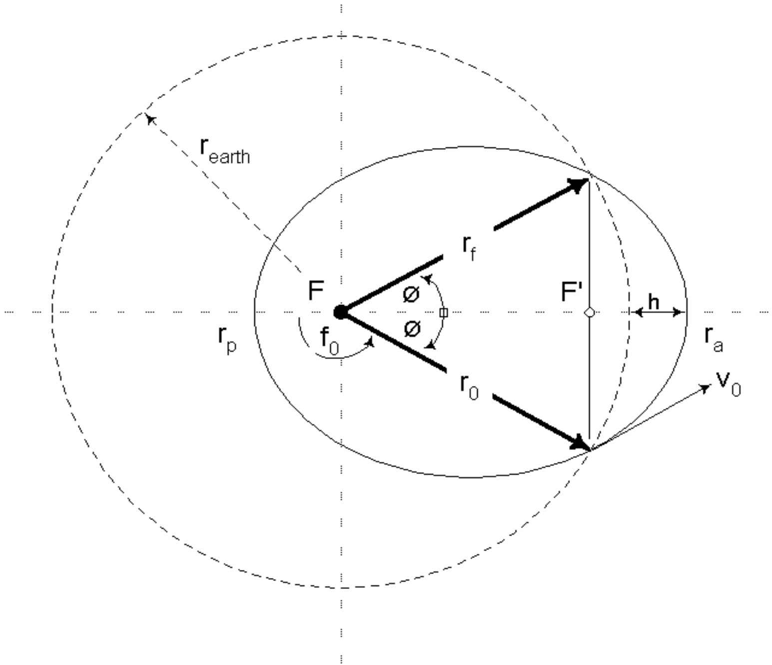


Figure 1 Ballistic trajectory

The following are evident from the geometry.

(1a) $$f_0 + f_f = 2\pi$$

(1b) $$E_0 + E_f = 2\pi$$

(1c) $$r_\oplus + h = r_a$$

(1d) $\Delta f = 2\phi$ = the range angle

As with the minimum energy Lambert problem, the optimal ballistic trajectory is when the chord between the initial and final position vectors goes through the empty focus of the ellipse.

(2) $$2a = r_0 + r_f$$

because the minimum energy flight path has the smallest a, from

(3) $$E = -\frac{\mu}{2a}$$

From the geometry, and symmetry, r_f is the semi-parameter

(4) $$r_f = p = a(1-e^2) = r_\oplus \sin\phi$$

And from (2) and (4),

$$2a = r_0 + p$$

(5) $$a = r_\oplus \frac{1+\sin\phi}{2}$$

The true anomaly of the launch point f_0 can now be found from the trajectory equation and substituting for e from (3)

(6) $$r_\oplus = \frac{a\left(1-e^2\right)}{1+e\cos f_0}$$

Kepler's equation can be used to determine the time of flight.

(7a) $$n\left(t_0 - t_p\right) = E_0 - e\sin E_0$$

(7b) $$-\left[n\left(t_f - t_p\right) = E_f - e\sin E_f\right]$$; adding (7a) and (7b)

(7c) $$\Delta t = \frac{2}{n}\left[\pi - E_0 + e\sin E_0\right]$$

An optimum eccentricity for the mission can now be calculated with a little effort.

$$r_0 = \frac{p}{1 + e\cos f} \;\; ; \;\; p = a(1 - e^2)$$

solving for a,

$$a = \frac{r_0(1 + e\cos f_0)}{1 - e^2}$$

Taking the derivative of this function for a with respect to e,

$$\frac{da}{de} = \frac{r_0\left(e^2\cos f_0 + 2e + \cos f_0\right)}{\left(1 - e^2\right)^2} = 0$$

Using the quadratic formula to solve for e

$$e = \frac{-2 \pm \sqrt{4 - 4\cos^2 f_0}}{2\cos f_0} = \frac{-1 \pm \sin f_0}{\cos f_0}$$

Thus, the optimum eccentricity is the simple formula that is a function of the initial true anomaly only.

(8) $$e = \frac{\sin f_0 - 1}{\cos f_0}$$

Discussion

The Hohmann Transfer is a ballistic trajectory. Generate a plot of the Hohmann Transfer from Earth to Mars, and overlay this with the geometry of the ballistic trajectory. Transfer coordinates to this new system and develop a set of equations that define the parameters in terms of the original set of orbital elements.

12. Hyperbolic Orbits

Interplanetary missions often involve several hyperbolic trajectories: Escape from Earth orbit, intermediate planet fly by(s) to gain speed or change direction, and planet capture at the target planet.

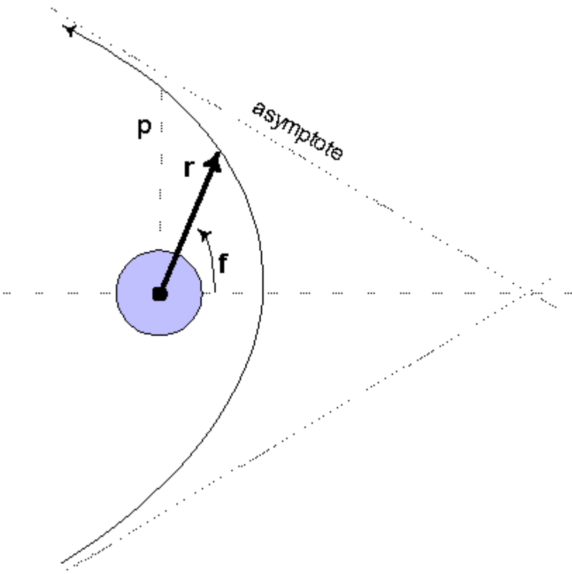


Figure 1 Hyperbolic Fly By

An important relationship for hyperbolas comes from an inspection of the energy integral (notice E > 0 for hyperbolic flight paths).

(1) $$E = \frac{v^2}{2} - \frac{\mu}{r} = -\frac{\mu}{2a}$$

As $r \to \infty$, $v^2 \to 2E$ and so

(2) $$v_\infty^2 = -\frac{\mu}{a}$$

Another important equation for hyperbolas starts with the trajectory equation (which works for all conic sections), rearranged

(3) $$\frac{p}{r} = 1 + e\cos f$$

Again, as $r \to \infty$, $e\cos f \to -1$.

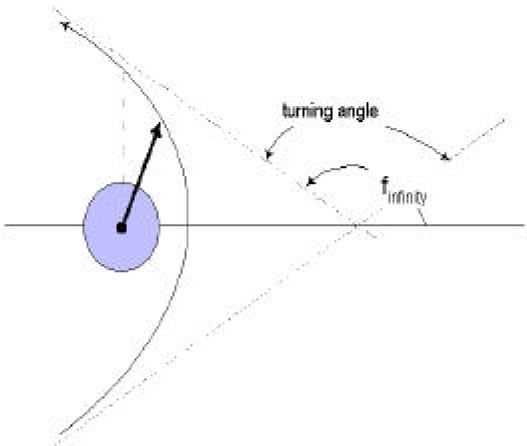


Figure 2 Turning Angle

The true anomaly as $r \to \infty$ approaches an asymptote of the hyperbola,

(4) $$f_\infty = \delta + \psi$$

where δ is the <u>turning angle</u>, i.e. the change in direction of the s/c as a result of the planet fly by. Also,

(5) $$f_\infty = \frac{\delta}{2} + 90° \Rightarrow \cos f_\infty = -\sin\frac{\delta}{2}$$

Substituting this value into (3)

(6) $$\sin\left(\frac{\delta}{2}\right) = \frac{1}{e}$$

Some more important characteristics are used in orbital mechanics to study hyperbolic fly bys.

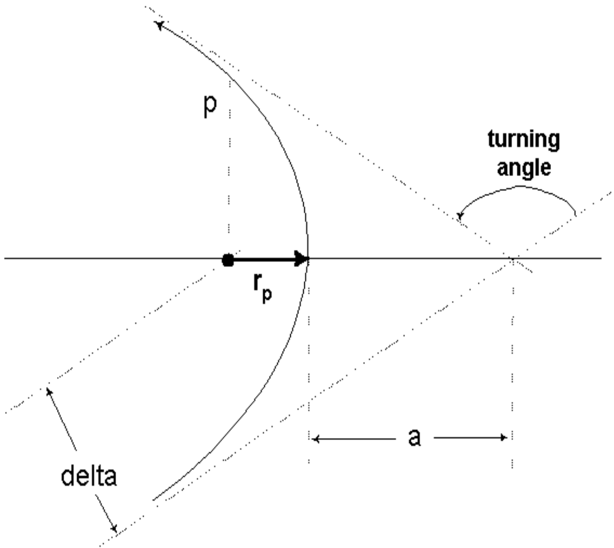


Figure 3 Targeting a flyby height

This diagram illustrates some important parameters

(7) $$h = r_p v_p = \Delta v_\infty$$

(8) $$e = 1 + \frac{r_p v_\infty^2}{\mu}$$

In making these calculations it is <u>important</u> to remember the following characteristics of hyperbolas

$$e > 1$$

$$a < 0$$

$$E > 0$$

As long as you keep the signs straight, the universal equations that work for all conic sections can be used for hyperbolas.

$$r = \frac{p}{1 + e\cos f}$$

$$r_p = a(1 - e)$$

$$p = a(1 - e^2)$$

$x = r\cos f$ and $y = r\sin f$

$$E = \frac{v^2}{2} - \frac{\mu}{r} = -\frac{u}{2a}$$

$$M = n(t - t_p)$$

There is also the equivalent of an eccentric anomaly for hyperbolic orbits, called the hyperbolic anomaly, F.

(9) $$M = e\sinh F - F$$

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

(11) $$y = (-a)\sqrt{e^2 - 1}\sinh F$$

(12) $$r = (-a)\left(e\cosh F - 1\right)$$

Discussion

The mean anomaly for ellipses has a geometric corollary for nearly circular orbits, as the angle from the empty focus. Determine if such a relationship exists for parabolic orbits, and the values of eccentricity for which this is valid.

13. Velocity Diagrams

The key to understanding the hyperbolic escape trajectory a s/c takes upon leaving Earth is the relative velocity of the Earth and the s/c. The easiest way to keep it all straight is by drawing diagrams. Assume a Hohmann Transfer as in section 9, with the same convention for V+, which is the heliocentric velocity at periapse on a transfer ellipse going say to Mars.

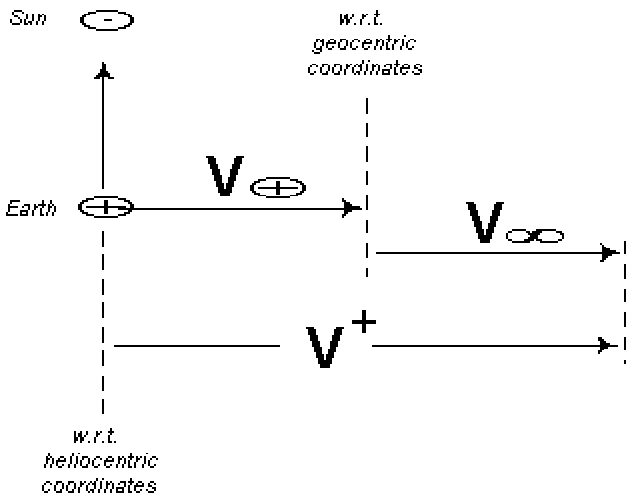


Figure 1 Relative Velocity Diagram

Figure 1 is the diagram for a s/c going to an <u>outer planet</u>, which means the velocity on the transfer ellipse will have to be faster than Earth. Consequently the $v_{\infty/\oplus}$ (v-infinity with respect to Earth) has to be in the same direction as the Earth's motion, of a magnitude

(1) $$v_{\infty/\oplus} = v^{+} - v_{\oplus}$$

Where v^{+} is the velocity at periapse on the heliocentric transfer ellipse, and $v_{\oplus}$ is the velocity of Earth at that point in its orbit.

Now, knowing $v_{\infty/\oplus}$ it is possible to solve for the velocity at periapse (where the thrust occurs) with respect to Earth $v_{p/\oplus}$, assuming r_p is provided as a mission parameter.

(2) $$E = \frac{v_{\infty/\oplus}^2}{2} = \frac{v_{p/\oplus}^2}{2} - \frac{\mu_\oplus}{r_{p/\oplus}} \Rightarrow v_{p/\oplus}$$

Then the total thrust that must be applied to begin the interplanetary trajectory from a low Earth orbit is

(3) $$\Delta v = v_{p/\oplus} - v_\oplus$$

The configuration of the relative velocity diagram changes if the mission is from Earth to an inner planet, say Venus.

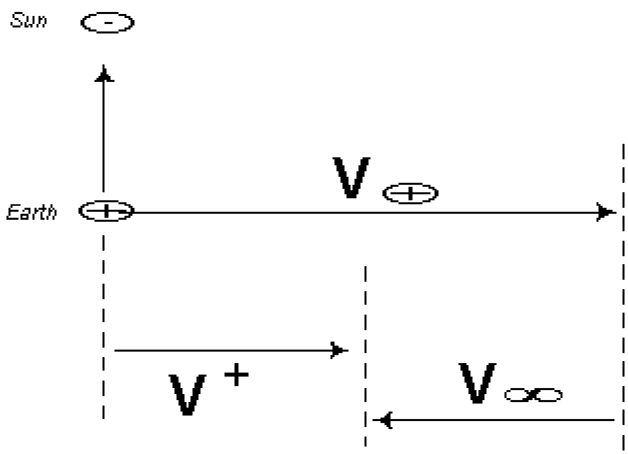


Figure 2 Going to an inner planet

Now the equations are, by inspection,

(4) $$v_{\infty/\oplus} = v_\oplus - v^+$$

(5) $\Delta v = v_{\oplus} - v_{p/\oplus}$

The importance of drawing these diagrams becomes evident when looking at the diagrams for planet capture on a hyperbolic path.

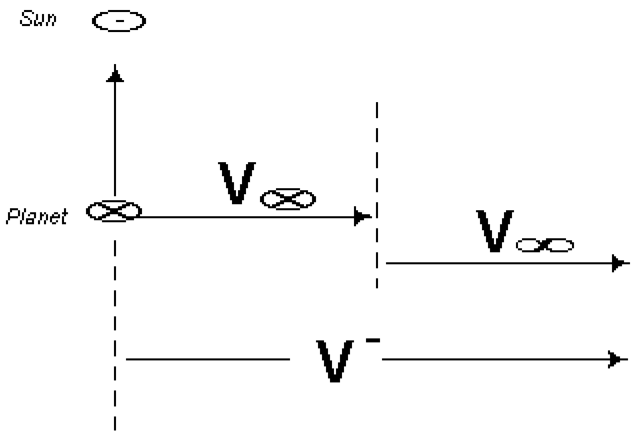


Figure 3 Hyperbolic approach to an inner planet

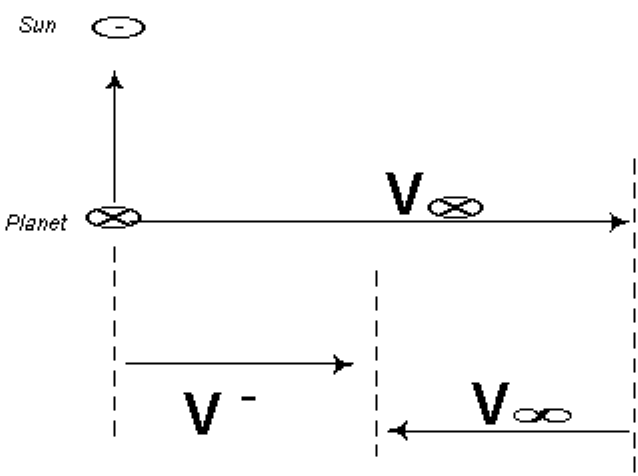


Figure 4 Hyperbolic approach to an outer planet

The equations are different for these planet capture hyperbolas. This time v_∞ is known and the approach geometry is needed. Usually Δ is specified as a mission parameter and the orbital elements of the approach hyperbola are solved for.

(6a) $$E = \frac{v_{\infty/\otimes}^2}{2} = -\frac{\mu_\oplus}{2a} \Rightarrow a$$

(6b) $$r_p = a(1-e) \Rightarrow e$$

(6c) $$\sin\left(\frac{\delta}{2}\right) = \frac{1}{e} \Rightarrow e$$

(6d) $$h = r_p v_p = \Delta v_\infty$$

where v_p (the velocity at periapse) comes from the relative velocity diagram.

Discussion

Create a set of relative velocity diagrams for a satellite going from near Earth orbit to geosynchronous orbit and vice versa.

14. Planet Fly By Geometry

Consider a mission from Earth to Mars. As the s/c approaches Mars at apoapse of its Hohmann Transfer ellipse, it's velocity is in the same direction as Mars, but it's magnitude is less.

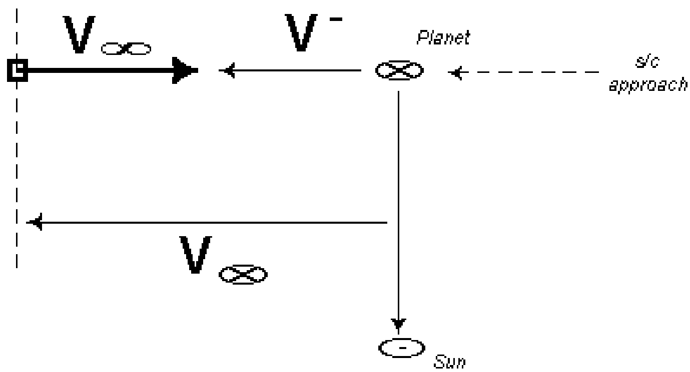


Figure 1 Relative velocities on Mars approach

As the s/c flys by Mars, the gravity changes the direction and magnitude of the s/c from **V- V+** ~ if the fly by is on the sunlit site of the planet.

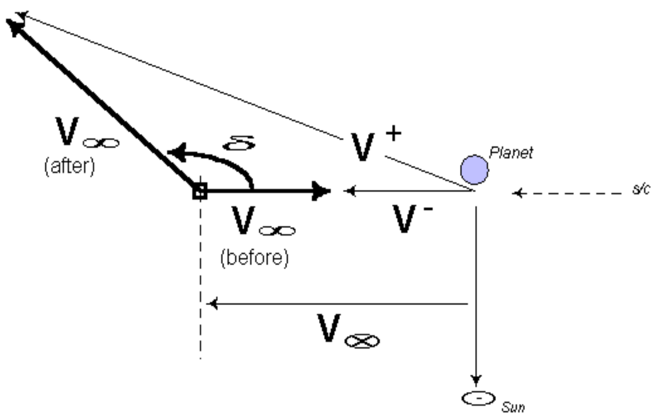


Figure 2 Fly By on the sunlit side of an outer planet

The key is that the v_∞ vector is rotated by the turning angle δ as shown, about a fixed point in the relative velocity diagram. The turning angle is in the opposite direction for a fly by on the dark side of a planet.

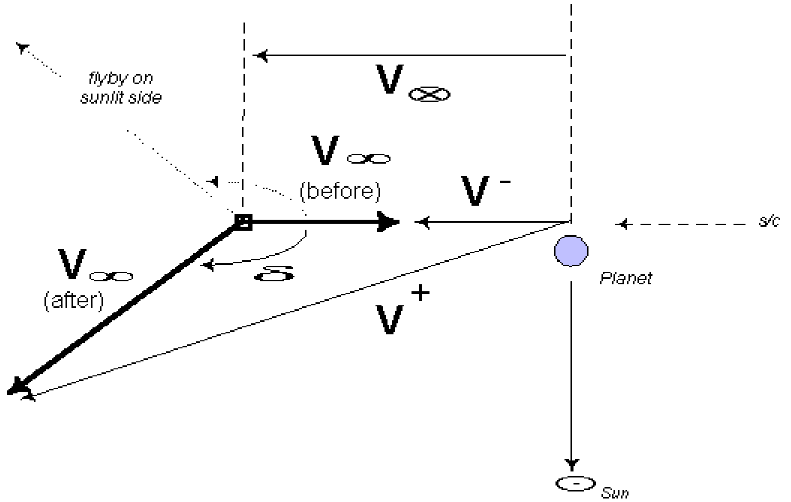


Figure 3 Fly By on the dark side of an outer planet

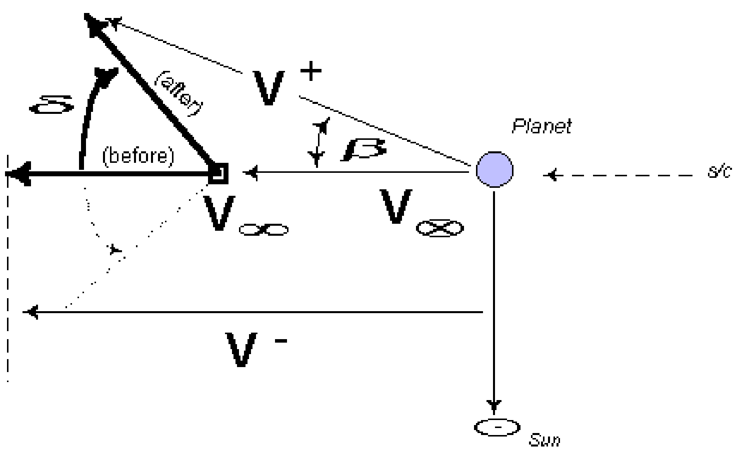


Figure 4 Fly By of sunlit side of an inner planet

The velocity vector is rotated about a different point for an inner planet flyby. The geometry is straightforward.

(1) $$v_\theta = v_\oplus - v_{\infty/\otimes} \cos(\pi - \delta)$$

(2) $$v_r = v_{\infty/\otimes} \sin(\pi - \delta)$$

(3) $$v^+ = \sqrt{v_\theta^2 + v_r^2}$$

(4) $$\beta = \arctan\left(\frac{v_r}{v_\theta}\right)$$

This method of mission planning follows implicitly what is called the patched conic method. For a mission, say from Earth to Mars, there are three problems that can be considered separately:

(i) a hyperbolic escape trajectory, when Earth is the central body

(ii) a heliocentric transfer ellipse when the Sun is the central body

(iii) a hyperbolic capture trajectory, when Mars is the central body

In each case the central body has the greatest influence upon the s/c by many orders of magnitude, so that other forces can be ignored. Technically speaking, the other major bodies exert third body forces upon the s/c and these forces are included in precise numerical models of the trajectory. However, for overall strategizing the patched conic is good, and it gives a thorough conceptual understanding of the forces involved the milestones of the mission itself.

The key to understanding the whole picture – i.e. to assembling the three pieces in a seamless mission plan – is the heliocentric velocities because they exist both in the main transfer ellipse and in the hyperbolic trajectories, where it's the velocity at infinity.

The place where the conic sections intersect is called the planet's sphere of influence. This is where the force of gravity from the sun is of the same order of magnitude as the force from the nearby planet. This is a very difficult region to compute accurate trajectories even with numerical methods, and will not be considered here further.

After a planet fly by the spacecraft begins a new elliptical heliocentric trajectory as soon as it leaves the planet's sphere of influence and the sun is its central body. Finding this new flight path is similar to the initial orbit determination problem.

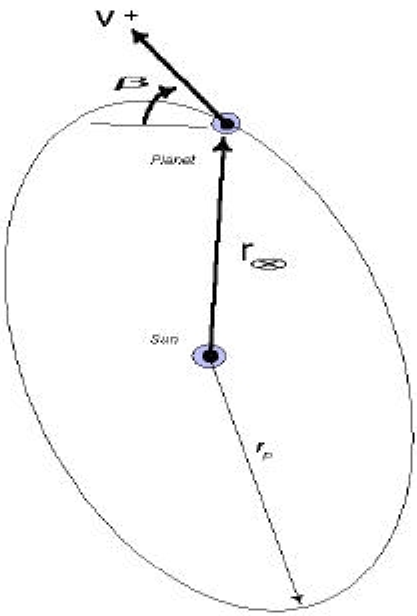


Figure 5 Heliocentric trajectory after a fly by

$$E = \frac{\left(v^+\right)^2}{2} - \frac{\mu_\odot}{r_\otimes} = -\frac{\mu_\odot}{2a} \Rightarrow a$$

$$h = r_\otimes v^+ \cos\beta = \sqrt{\mu_\odot p} \Rightarrow p$$

$$p = a\left(1 - e^2\right) \Rightarrow e$$

$$r = r_\otimes = \frac{p}{1 + e\cos f} \Rightarrow f$$

Discussion

Consider that the central body in the last diagram is Earth and the planet is the equatorial bulge of Earth. Use this to model the ballistic trajectory going from the northern to southern hemisphere, perpendicular to the equator. This is a Three Body Problem. Knowing the value of J2, find the perturbation on the orbit in the Two Body model. Assign a mass to the third body, sufficient to simulate the same orbit, in the Three Body model. Model the free flight part of the trajectory as a patched conic – e.g. as a hyperbolic capture trajectory, patched to a geocentric ellipse. Determine how much the fly by of the equatorial bulge changes the flight path.

15. Gravity Field Force

The most important equation in orbital mechanics, the Two Body Equation,

$$\ddot{\vec{r}} = -\frac{\mu}{r^3}\vec{r}$$

was derived from Newton's Law of Gravity. This same equation can be derived another way, using the concept of the <u>gradient of the potential</u>, U. That is, for a point mass

(1) $$U_{point\,mass} = \frac{\mu}{r}$$

which is the equivalent, in dynamical theory, of potential energy. The radius vector in an inertial system is

(2) $$\vec{r} = x\hat{i} + y\hat{j} + z\hat{k}$$

From which the magnitude of the radius vector is

(3) $$r = \sqrt{x^2 + y^2 + z^2}$$

Assuming U is a vector, it is possible to take it's gradient,

(4) $$\nabla U = \frac{\partial U}{\partial x}\hat{i} + \frac{\partial U}{\partial y}\hat{j} + \frac{\partial U}{\partial z}\hat{k}$$

Substituting (1) for U and taking the partial derivatives,

(5) $$\nabla U_{point\,mass} = -\frac{1}{2}\frac{\mu}{\left(x^2 + y^2 + z^2\right)^{3\backslash 2}}\left[2x\hat{i} + 2y\hat{j} + 2z\hat{k}\right]$$

Which is a statement of the Two Body Equation, and equals $\ddot{\vec{r}}$.

Thus, acceleration of a Two Body orbit is the gradient of the potential

(6) $$\nabla U_{point\,mass} = -\frac{\mu}{r}\vec{r} = \ddot{\vec{r}}$$

DISCUSSION: For non uniform masses it is necessary to define U as the sum of contributions from many components of the central body. This is done using a volume integral, and Lagrange Polynomials. That is,

$$U = \iiint_{volume} \frac{G}{R}\,dm[\quad]$$

The Lagrange Polynomial is the quantity in brackets. The J2 effect for the equatorial bulge has terms in the expansion that have a strong resemblance to $\dot{M}$ and $\dot{\omega}$; i.e. $\left(3\cos^2\phi - 1\right)$ and $\left(5\cos^2\phi - 1\right)$ described earlier for orbital perturbations by J2. Consider the possibility that specific terms of the Lagrange expansion might have specific influences upon the orbit around a central body. This could happen only if there were a mechanism by which any reference system used (i.e. the spatial origin is the center of the central body) were made inertial for use in these calculations for a non uniform body. Is this what the f and g functions accomplish?

19. Gravitational Shock Wave

It is now time to develop a numerical test to show if the f and g functions really do represent definite forces in the gravitational scheme of things. Assume that they DO exist, and form a model at the largest scale for which good data is known – i.e. for motion in the Solar System. The theory developed hence implies that one of the functions emanates from the inner planets of the solar system; the other from the outer.

In the terminology of wave dynamics, the f and g functions generate wave-fronts which should exhibit interference phenomena at their boundary. This boundary is the region between the inner and outer planets – occupied by the asteroids (and comets, many of whose focus is in this region); although the invariant plane analysis implies it is closer to Mars' orbit.

In any event, assume that some strong gravitational undertow exists in the region of Mars. When two independent wave-fronts meet, they form a shock wave: i.e. a new wave-front different from its two constituents, that moves faster than either of the other two. In other words, this gravitational anomaly will act to accelerate a spacecraft out of the Solar System. If that is the case, then a numerical model of the Earth to Mars trajectory should show some result of this phenomenon.

Such a model was constructed, and a formal technical paper describing how it works follows. The routine found a noticeably faster mission to Mars than the Hohmann; tentative proof of this f and g theory. The complete source code is attached (there is MUCH room for improvement). Perhaps fine tuning will generate a trajectory that will accelerate spacecraft even faster; on the order of comet like paths that go far beyond our Solar System at velocities not achievable using simple thrust engines.

For instructional purposes, assignments can be made from each Section of the book to create the different subroutines used in the computer simulation – to solve Kepler's equation, convert from heliocentric to geocentric coordinates, etc.. – so that by the end of the semester, each student can then form this optimization of the complete trajectory.

OPTIMIZING THE EARTH TO MARS TRAJECTORY

William H. Clark II, P.E., MSE

ABSTRACT

The interplanetary trajectory from Earth to Mars is difficult to solve numerically. Typically a problem with four or more unknowns requires a genetic algorithm to solve, the problem being intractable to any other nonlinear optimization methods. This paper shows how an Earth to Mars trajectory with twenty five unknowns can be solved directly, without the use of any nonlinear optimization methods whatsoever. The basic method finds a trajectory which is either thirty days faster than a flight path using the same total thrust as a Hohmann Transfer (i.e. the minimum energy trajectory in two body motion) or a trajectory which has the same time of flight as the Hohmann but uses 10% less total thrust. This report shows how the optimization algorithm is constructed and how it arrives at a solution using a relatively compact body of code that solves the problem to 12 significant digits in seconds on a personal computer.

INTRODUCTION

The simplest and most efficient interplanetary trajectory is the Hohmann Transfer, the usual standard to which solutions are compared. The Earth to Mars trajectory, the topic of this report, had no analytical solution. The problem must be modeled on the computer and numerically integrated. A typical NASA mission has four thrusts: the initial thrust at Earth plus four Trajectory Correction Maneuvers (TCMs). At present, a simple trajectory with just two thrusts can be solved only with the benefit of a nonlinear optimization routine, whereas anything more complex must be solved in stages by genetic algorithm methods.

The patched conic method forms the basis for all interplanetary trajectory studies, by which the flight path is set up in specific configurations according to the forces acting upon the spacecraft (s/c). When the s/c is close to Earth, the central body is Earth and the Sun is a third perturbing body. This changes at the Sphere of Influence (SOI), after which time the Sun is the central body and Earth and Mars are perturbing bodies. (The SOI is the point where the force from the central body and a third body are equal in magnitude.) Then at Mars' SOI, the central force is Mars and the Sun is a third perturbing body. The transitions between planet centered and heliocentric coordinates is more than a convenience reference, but also the most accurate way to represent the forces from all perturbing bodies.

The trajectory to Mars from Earth is optimized in this paper without the benefit of any nonlinear optimization method. The two point boundary value problem with free time and position at both end points is separated into two independent problems with a common, fixed end point, approximately at conjunction. The algorithm then considers variations from the Hohmann Transfer. It converges directly to the solution. The method is accurate and fast, and generates an optimal solution that is either 10% better than the Hohmann in terms of total thrust or total time of flight.

The algorithm is compact and converges rapidly, the entire optimization taking less than thirty seconds on a 333 MhZ personal computer. This is many orders of magnitude faster than other methods of solution working on problems of a comparable difficulty. A total of 25 variables are optimized in this method. Such an algorithm would be useful for on board flight adjustments by a semi autonomous s/c, being able to perform real time calculations that currently can only be performed by mainframe computers at flight control on Earth. Also, many additional degrees of freedom can be added to this basic algorithm, and the entire problem then solved by third party nonlinear optimization methods. That is, for a problems with more than twenty five variables (e.g. a 3D simulation), this algorithm generates a nominal solution much closer to the optimum than the usual starting point, the Hohmann Transfer. Finally, the solution method is not abstract but easily visualized conceptually, giving the model and the algorithm a realistic aspect that makes it easy to imagine the forces, the bodies, and the overall characteristics of the entire problem. By comparison, the genetic algorithm is wholly statistical and uses a completely random way to arrive at the solution.

ESTABLISHING THE BENCHMARK

As the computer model was being constructed, starting from a simple Hohmann Transfer between Earth and Mars in circular orbits, a standard was used to test the results before going on to the next, more complicated elaboration of the model. The standard used was the Hohmann Transfer, by which the s/c makes the transit from Earth to Mars in exactly 180 degrees of heliocentric longitude. Two thrusts are required, on at either end of the trajectory:

dV1 = 2.944 km/sec
dV2 = 2.649 km/sec
total = 5.593 km/sec
time = 260 days

The model solves the 2001 Earth to Mars trajectory, starting in a 200 km parking orbit at Earth and ending in a 100 km parking orbit at Mars. All motion is assumed coplanar with Earth and Mars are in their true elliptical orbits. Table 1 shows a range of values for a given TCM at conjunction.

Table 1 Trajectory Parameters versus the Hohmann Transfer

DeltaV, km/sec	time, days	total deltaV, km/sec
1.2	214.224	6.764
1.1	216.133	6.511
1.0	218.019	6.308
0.9	220.063	6.157
0.8	222.766	5.779
0.7	225.131	5.645
0.6	228.339	**5.625**
0.5	231.510	5.283
0.4	235.749	5.381
0.3	240.580	5.202
0.2	247.730	5.131
0.1	**262.190**	5.157

NOTE: The Hohmann Transfer time is 260 days and 5.593 km/sec

THE MID COURSE CORRECTION

The unique aspect of the algorithm is that a major thrust is done at conjunction, or about half way to Mars from Earth. (Use of “conjunction” throughout this report is meant to signify an approximate location, at about 90 degrees from start in the trajectory, which coincides with the Earth-Mars conjunction in this particular situation.) There are several reasons why conjunction was chosen:

- The s/c travels slowest in the second half of the trajectory, so the greatest benefit from a major thrust would be if this thrust were applied at conjunction
- A thrust applied at a true anomaly of 90 degrees (e.g. conjunction for this particular problem is at a true anomaly of 94 degrees) acts to elongate the elliptical orbit. A thrust applied at any other point in the trajectory acts to rotate the orbit in space. An elongated orbit is by far the best configuration as it causes the transfer orbit to intersect the path of Mars for a shorter time of flight.
- The bi-elliptic transfer orbit is actually more efficient than a Hohmann, but is usually not used because it has a much longer time of flight. In this situation, separating the problem at conjunction and applying a large thrust there makes the solution into a bi-elliptic transfer orbit, where the geometry of both parts is favorable to the goal of finding a faster, more efficient trajectory.

BASIC FORMULATION

The flight path from Earth to Mars has seven specific reference points, as follows:

1. A 200 km Earth parking orbit
2. The beginning of the interplanetary trajectory, near Earth but not necessarily at the initial parking orbit
3. Earth's SOI
4. The Earth-Mars conjunction
5. The end of the interplanetary trajectory, near Mars but not necessarily at the final parking orbit
6. A 100 km Mars parking orbit

The objective of the analysis to optimize this fixed flight path between two points of constant energy, i.e. the designated parking orbits around Earth and Mars. The mission profile allows for a maximum of five thrusts. The optimal path begins with a small Hohmann Transfer from Earth parking orbit to a slightly higher orbit, then a main thrust to escape

Earth gravity, followed by a mid-course correction at conjunction, and a small Hohmann Transfer at Mars into the final parking orbit around Mars.

It is assumed that a Hohmann Transfer can reach the intermediate "targeting" orbits most effectively, and that the thrusts for this planet centered trajectory can be approximated using two body equations of motion. The s/c enters the circular targeting orbit, from which it begins the integrated interplanetary trajectory with a large instantaneous thrust.

The flight path is integrated between the respective targeting orbit end points with three interruptions: one at each planet's SOI, and a third at conjunction. Within the SOI the integration uses planet centered coordinates and the sun as a perturbing third body. Otherwise, the coordinates are heliocentric with the motion of the s/c perturbed by Earth and Mars.

Throughout this analysis, all thrusts are aligned to the direction of motion. This is the most efficient thrust, transferring all energy to velocity and not wasting any energy on changes in the shape or orientation of the orbit itself. In so doing, all three major tangential thrusts increase the ellipticity of the orbit (for reasons noted earlier); again, to get the most out of the thrust in velocity.

EQUATIONS OF MOTION

The equations of motion are the standard Newtonian formulation for a gravitation force between two bodies. The integration is divided into two problems, both starting with the same state vector (coordinates and position) at or near conjunction. The motion of the s/c is integrated back in time to reach Earth (all forces are central forces, so integration with a negative time step is OK), with a variable magnitude and direction to optimize both trajectories, independently.

The computer model is for elliptical, coplanar motion of the primaries using the J2000 ephemeredes. The position of the primaries – i.e. Earth and Mars – is propagated using two body dynamical equations, with the inclination of Mars' orbit to the ecliptic set to zero.

The integration is performed by a Runge-Kutta variable step (7/8) integrator set to a tolerance of 10E-12, to give a consistent accuracy of 12 to 13 significant digits. The integration step is not altered externally with the exception of when the trajectory nears the SOI so the integration can be stopped as near to the SOI as possible to transfer the coordinates into a new reference system.

BOUNDARY VALUE PROBLEMS

The computer algorithm is set up to optimize each half of the flight path independently. A fixed thrust is applied at conjunction, then the minimum thrust to reach parking orbit is determined. The “terminal” thrust is assumed to have three components: a large thrust to circularize at the targeting orbit, and two small thrusts to reach the final parking orbit via the Hohmann Transfer. Typically, the large thrust is combined with one of the two small thrusts, into one instantaneous thrust.

The numerical implementation of this strategy necessitates the creation of a unique mapping in the vicinity of the target. Since the objective is to find a global minimum, it is essential that every thrust value applied at conjunction generates a total thrust to reach parking orbit. This total thrust need not be accurate to 12 significant digits except near the actual minimum. However, it must have a constant slope downward to the minimum from all directions so that a one dimensional line search can find the minimum easily. A cross section of the mapping created is shown in Figure 1.

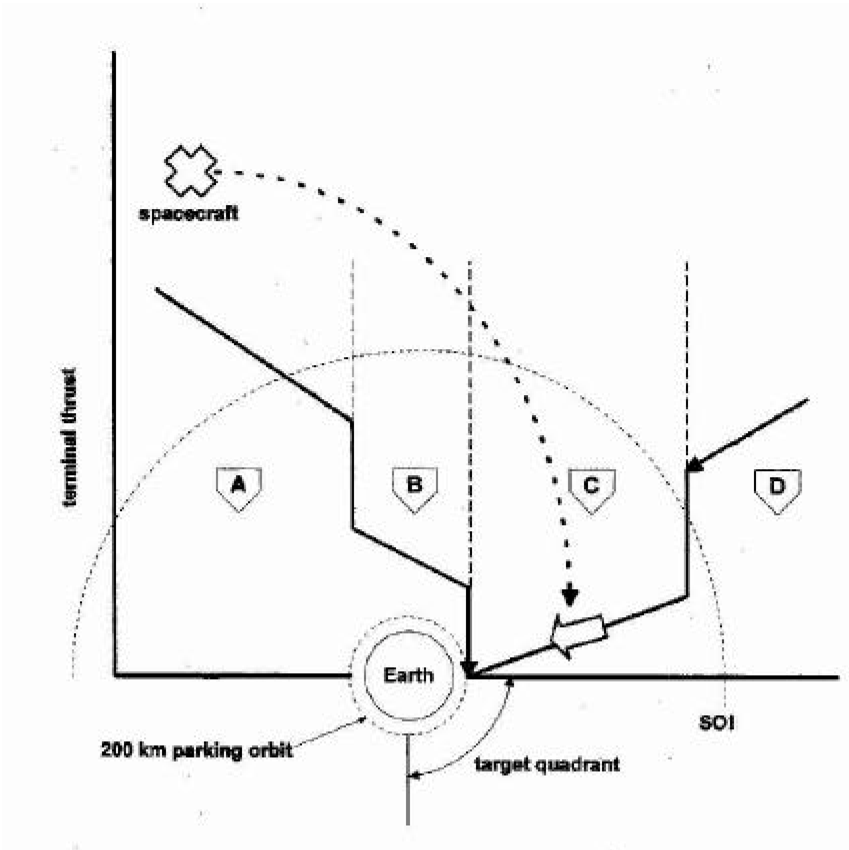


Figure 1 Mapping the Earth Escape trajectory

The energy/thrust "well" simulated by the algorithm returns a terminal thrust in such a way that it drives the final endpoint to the quadrant noted in Figure 1. This is a point on the far side of the planet from which the s/c approaches, located at or near the final parking orbit itself.

NUMERICAL OPTIMIZATION

The first objective of the algorithm is to find the initial conditions for Earth, Mars, and the s/c. It has already been established that the integration will begin at conjunction, for the specific case of the first conjunction of the J2000 ephemeredes. See Figure 2.

The hypothesis to be tested assumes that the conjunction to Earth trajectory on a Hohmann Transfer ellipse is already optimal, so it is only necessary to find the initial position of Earth, i.e. 94 degrees before conjunction. To find the starting point of the s/c, it's path on the transfer ellipse is found at 96 days (the time for Earth to reach conjunction in the Hohmann Transfer) after periapse, i.e. when its position is coincident with Earth.

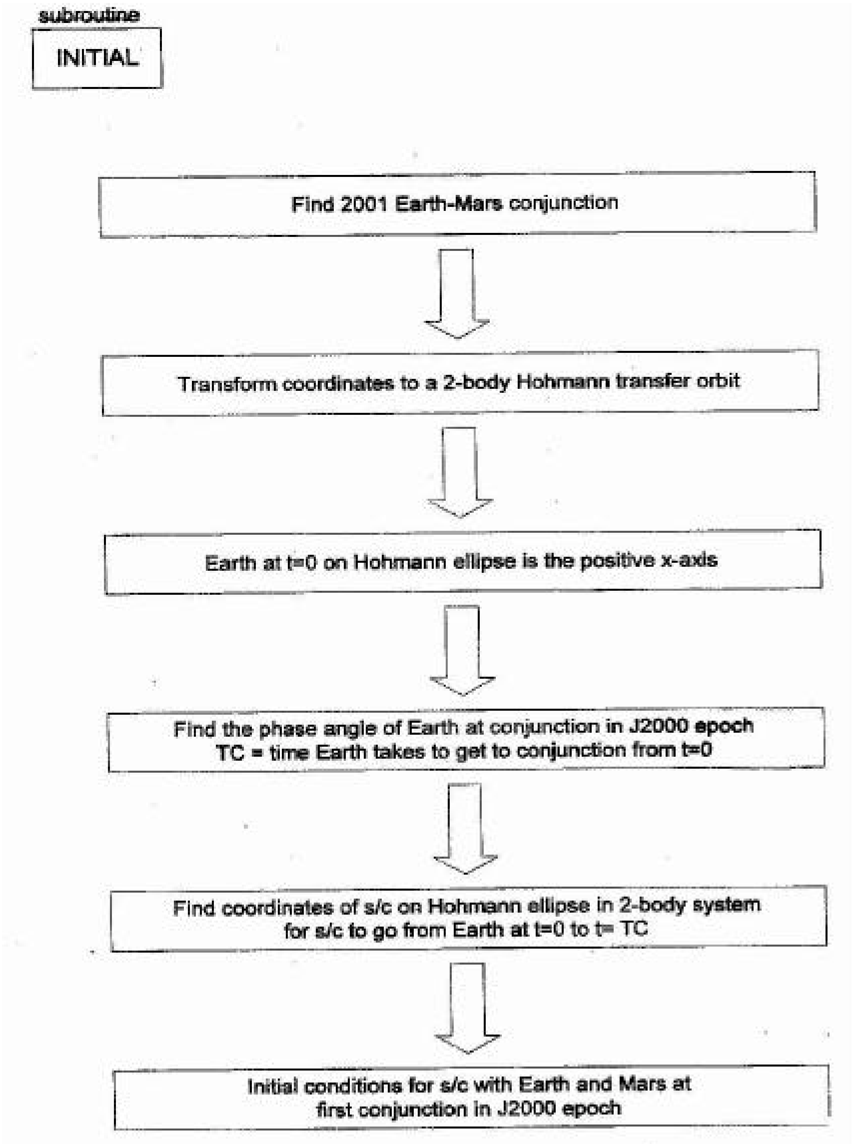


Figure 2 Determining the Initial State

The integration is done in two parts: back in time from conjunction to Earth and forward in time from conjunction to Mars. Earth and Mars are incremented by their two body equations of motion, along elliptical coplanar paths using mean motion. The s/c is integrated with either the Sun or a planet as the central body, and the other major bodies perturbing.

As indicated in Figure 3, the trajectory to Earth is optimized only once. Both endpoints of the trajectory are known (i.e. conjunction and a 200 km parking orbit around Earth), so once the minimum thrust trajectory is found that half of the problem is solved.

The conjunction to Mars part of the trajectory has one fixed endpoint at conjunction and the other point variable. The time of flight to Mars is a function of the thrust applied at the mid-course correction at conjunction. The starting position of Mars is varied so that the s/c intersects Mars. This is the part of the trajectory that is most susceptible to improvement and is therefore more closely evaluated in the optimizing routine.

Conjunction to Earth

The technique for optimizing the conjunction to Earth flight path is shown in Figure 4. The initial state of the s/c is integrated back in time to Earth’s SOI. Then the coordinates are changed from heliocentric to geocentric and the path is integrated until the range from Earth is increasing.

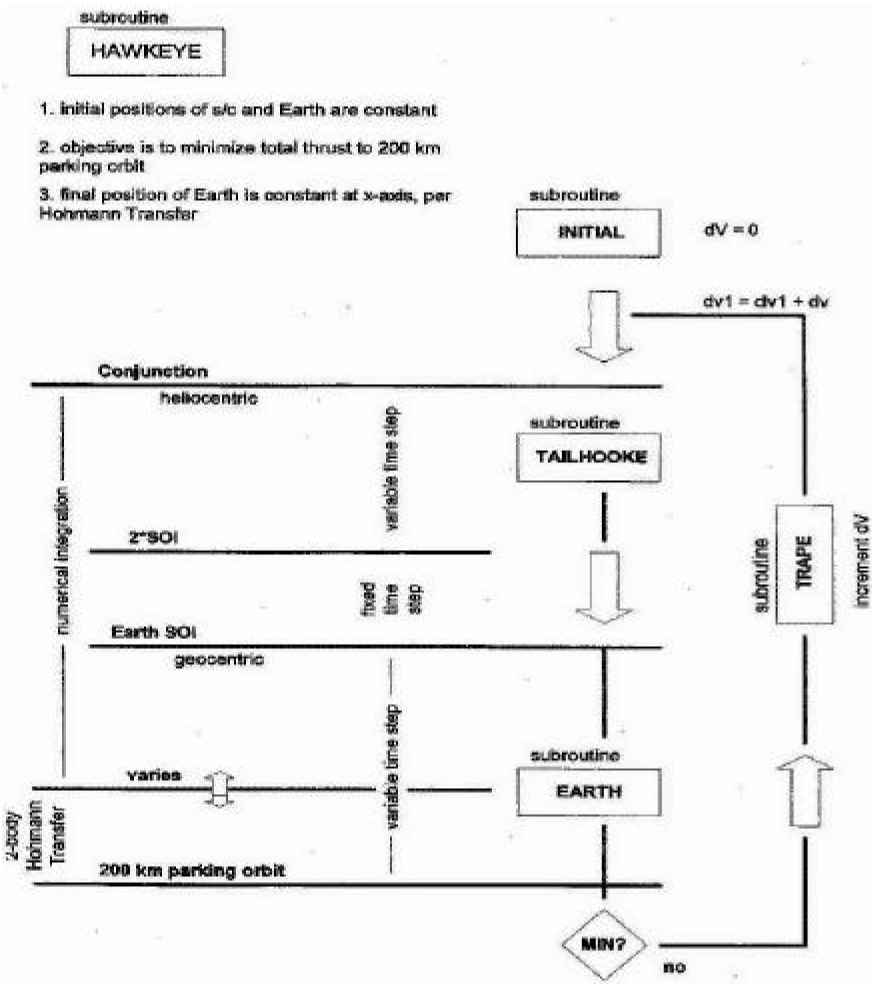


Figure 4 Targeting Earth from Conjunction

The integration is done with a Runge-Kutta 7/8 variable step integrator. At each step the thrust needed to reach the 200 km final parking orbit is calculated (i.e. the orbit is circularized, then a small Hohmann Transfer is done to the final parking orbit). This is shown in Figure 5, a representation of the final, optimal trajectory near Earth.

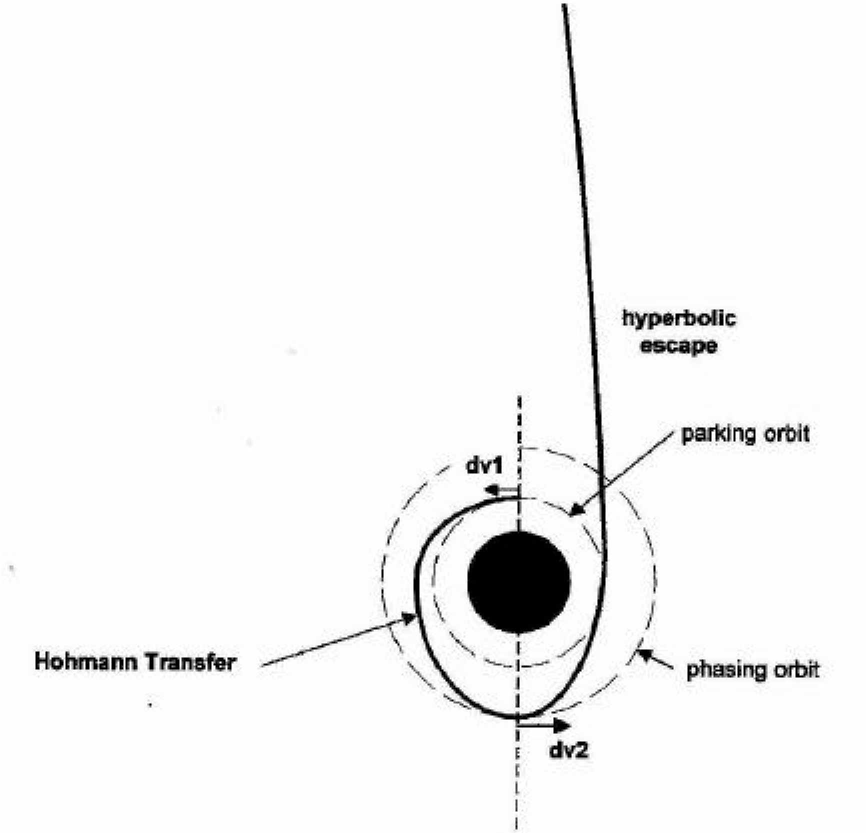


Figure 5 Earth Escape Geometry

It is assumed that the integrator stops for any significant changes in conditions on the s/c, thus no intermediate targeting orbits need to be considered. As the s/c nears Earth, the lowest total trust is saved in an array and when the range to Earth begins to increase, the loop is terminated and the minimum thrust scenario is retrieved from the array as the optimal trajectory.

After a single trajectory is optimized, the total thrust is compared to the total thrust of the previous run. If it is decreasing, the thrust at conjunction is decreased (and vice versa) and another trajectory is evaluated. The mapping described in Figure 1 is such that the global minimum is always in the direction of a negative gradient, so all the iterative loop must do is to conduct a one dimensional search until the minimum is reached within a specified tolerance.

Conjunction to Mars

The method by which the conjunction to Mars flight path is optimized is shown in Figure 6.

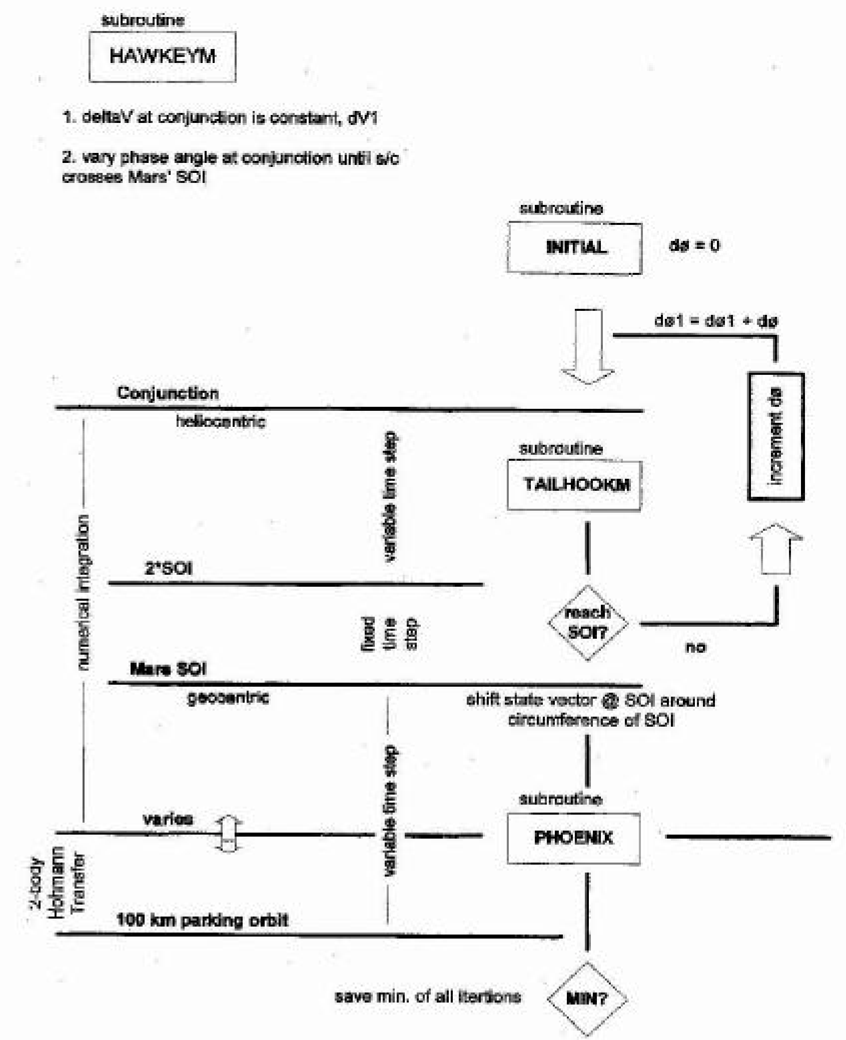


Figure 6 Targeting Mars

The first objective is to adjust the phase angle of Mars at conjunction so the s/c path intersects Mars' SOI. This is a highly nonlinear problem and it cannot be solved by direct method used for the trajectory to Earth. Instead, once the state vector of the s/c is known at SOI, it is assumed that a small TCM can be done in real time to target any point on the SOI in the immediate neighborhood. Modifying the tolerances in the code can make the solution exact, at the cost of perhaps doubling the computational time. This was not done in the distribution copy of the program so the program can run on a PC in less than 60 seconds to get the approximate solution. The maximum possible error is 4 hours in the total time of flight of 260 days (± .001% ~ this decreases by an order of magnitude with each iteration). The total thrust is not changed because of how the algorithm is structured.

CONCLUSION

The algorithm used to optimize the Earth to Mars trajectory is simple, direct, and has a rapid convergence to a global minimum. The code is adaptable in that (1) additional perturbations from other major planets can be added easily; (2) additional user defined features can be added to further minimize the time/thrust; e.g. by allowing a retrograde parking orbit at Mars for an additional gravity assist at either planet by allowing a lower fly by altitude than the respective parking orbits; (3) the routine is easily adaptable to parallel processing, as large blocks of code can be executed independently; and (4) the approximation made to the trajectory at Mars SOI – the only such approximation in the whole algorithm – is made at this point in the flight path because this is where, on actual missions, final approach adjustments are made to target a specific latitude and longitude, so the code can be easily modified to allow for this possibility in future versions.

REFERENCES

Bate, Mueller, White, “Fundamentals of Astrodynamics.” Dover Publications, 1971

Battin, Richard, “An Introduction to the Mathematics and Methods of Astrodynamics,” AIAA Education Series, 1999

Danby, J.M.A., “Fundamentals of Celestial Mechanics,” Willmann-Bell, 1992

Jefferys, William, Class Notes for Celestial Mechanics I at the Univ. of Texas at Austin, fall of 2000

Kaplan, Marshall, “Modern Spacecraft Dynamics and Control,” John Wiley and Sons, 1976

Roy, A.E. “Orbital Motion,” Adam Hilger, 1988

Vallado, David, “Fundamentals of Astrodynamics and Applications,” McGraw-Hill, 1997

Vaughan, R., Kailemeyn, P., Spencer D., Braun. R., “Navigation Flight Operations for Mars Pathfinder,” AAS Paper 98-145, February 1998


```
c    SUBROUTINE  ClarkBert

c     Integrated Lambert Transfers

c     Copyright 2000, William H. Clark II

c    All Rights Reserved
c     Last modified 6/3/00
c
      CALL gui
      STOP
      END
C
C
      SUBROUTINE TailHookm (x,t,deltaV,Vrel)
      implicit real*8 (a-h,o-z)
      common/output/tj
      common/const0/pi,tol,iflag,itarget,idim
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
      real*8 x(6),xm(6),x12(6),xg(6),tj(30,3)
      dt = +100d0
      do 75 n = 1,1500
        CALL RK78T (t,x,dt,tol,6)
        range = range2 (x,xm,x12,sense,Vrel)
        rhoM = dsqrt(xm(1)**2+xm(2)**2+xm(3)**2)
        rhoX = dsqrt( x(1)**2+ x(2)**2+ x(3)**2)
        cable1 = rhoM - soiM*50d0
        cable2 = rhoM - soiM*3d00
        IF (rhoX .GT. cable1) dt = +86400d0/1d0
        IF (rhoX .GT. cable2) dt = +86400d0/4d0
        IF (rhoX .GT. rhoM -soiM) THEN
```

```
            deltaV=dmod(datan2(x(2),x(1))-datan2(xm(2),xm(1)),2*pi)
            GOTO 78
          ENDIF
          range1 = helioTOgeo (x,t,4,xg)
          axis = datan2(xg(2),xg(1))
          zone = datan2(xg(5),xg(4))
          IF (t/86400d0 .GT. 250d0) THEN
            GOTO 78
          ENDIF
   75 continue
   78 return
      end
c
      SUBROUTINE catapult (beta1,beta2,deltaVi,deltaVf,tsip)
      implicit real*8 (a-h,o-z)
      common/output/tj
      common/constopt/alpha,beta,t1,t2
      common/const0/pi,tol,iflag,itarget,idim
      dimension tj(30,3)
c
      pi = dacos(-1.d0)
      tol = 1.d-13
      CALL INRK78
      idim = 2
c
      CALL constants
      CALL HawkeyM (beta2,deltaVi,dVmars)
      CALL constants
      CALL HawkeyE (beta1,dVearth,deltaf)
      tsip = (t2-t1)/86400d0
      deltaVf = ABS(dVearth) + ABS(deltaf) + ABS(deltaVi) + ABS(dVmars)
      tj(22,1) = tsip
      tj(22,2) = deltaVf
      RETURN
      END
C
      SUBROUTINE HawkeyM (beta2,deltaVi,dVmin)
      implicit real*8 (a-h, o-z)
      dimension x(6),xg(6),tj(30,3)
      common/output/tj
      common/constopt/alpha,beta,t1,t2
      common/const0/pi,tol,iflag,itarget,idim
c
      iflag = 1
      beta = beta2
      alpha = 0.d0
c
      do 2 i = 1,10
      CALL initial(deltaVi,x,t)
      CALL TailHookm (x,t,delta,Vrel)
      alpha = alpha + delta
      IF (ABS(delta) .LT. 1d-6) GOTO 3
    2 continue
c
```

```
    3 tmars = t
      tj(13,1) = x(1)
      tj(13,2) = x(2)
      tj(14,1) = x(4)
      tj(14,2) = x(5)
      tj(15,1) = tmars/86400.d0
      range2 = helioTOgeo (x,t,4,xg)
      tj(16,1) = xg(1)
      tj(16,2) = xg(2)
      tj(17,1) = xg(4)
      tj(17,2) = xg(5)
      CALL phoenix (t,Vrel,angle,dVmin)
      t2 = tmars + t
      RETURN
      END
c
      SUBROUTINE phoenix (t,Vrel,angle,dVmin)
c     level THREE diagnostics:  seek free return trajectory
      implicit real*8 (a-h,o-z)
      dimension xe(6),xm(6),xg(6),xtable(40,14),tj(30,3)
      common/output/tj
      common/const0/pi,tol,iflag,itarget,idim
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
c
      Rf = radM + rpM
      gamma = -pi/8.d0 + .0003d0
      do 4 j = 1,40
         gamma = gamma + pi/220.d0
         xtable (j,1) = -soiM*dsin(gamma)
         xtable (j,2) = -soiM*dcos(gamma)
         xtable (j,3) = 0.d0
         xtable (j,4) = 0.d0
         xtable (j,5) = Vrel
         xtable (j,6) = 0.d0
         xtable (j,7) = 0.d0
         xtable (j,8) = 0.d0
         xtable (j,9) = 0.d0
         xtable (j,10)= 0.d0
         xtable (j,11)= 0.d0
         xtable (j,12)= 0.d0
         xtable (j,13)= 0.d0
         xtable (j,14)= 0.d0
    4 continue
      gmsun = gmmars
      gmearth = 0.d0
      gmmars = 0.d0
      gmjupitr = 0.d0
      tmax = 10.d0*soiM/Vrel
c
      do 78 m = 1,40
        do 8 k = 1,6
    8     xg(k) = xtable (m,k)
        t = 0.d0
```

```
      dt = +1000.d0
      dVmin = 100.d0
      Vrel = xtable(m,5)
      do 75 n=1,1500
        CALL RK78T(t,xg,dt,tol,6)
        range = sqrt(xg(1)**2 + xg(2)**2 + xg(3)**2)
        IF (range .LT. radM) THEN
          deltaV = deltaV*2.d0
          GOTO 78
        ENDIF
        CALL velocity (xg,gmsun,range,Vrel,Rf,deltaV,V1,V2,V3)
        IF (deltaV .LT. dVmin) THEN
          do 10 k = 1,6
 10         xtable (m,k) = xg(k)
          xtable (m,10) = deltaV
          xtable (m,9) = gamma
          xtable (m,8) = range
          dVmin = deltaV
          xtable(m,11) = V1
          xtable(m,12) = V2
          xtable(m,13) = V3
          xtable(m,14) = t/86400d0
        ENDIF
        IF (range .GT. soiM*1.1d0) GOTO 78
        IF (t .GT. tmax) GOTO 78
 75     continue
 78 continue
    do 80 m = 1,40
    IF (xtable (m,10) .LT. dVmin .AND. xtable(m,8) .GT. radM) THEN
        tj(18,1) = xtable(m,1)
        tj(18,2) = xtable(m,2)
        tj(19,1) = xtable(m,4)
        tj(19,2) = xtable(m,5)
        tj(20,1) = xtable(m,14)
        tj(21,1) = xtable(m,12)
        tj(21,2) = xtable(m,11)
        tj(21,3) = xtable(m,13)
        dVmin = xtable(m,10)
        angle = xtable(m,9)
    ENDIF
 80 continue
    CALL constants
    RETURN
    END
C
    SUBROUTINE trapE (ax,step,dX,ttol,adelta)
    implicit real*8 (a-h,o-z)
C     beta angle is fixed >> find min. deltaV to Rpark
    CALL SG (ax,adelta)
c     print *, ax,adelta
  2 DO 3 k = 1,50
      IF (step .LT. ttol) GOTO 5
      bx = ax + step*dX
      CALL SG (bx,bdelta)
```

```
c         print *, bx,bdelta
        IF (bdelta .GT. adelta) GOTO 4
        ax = bx
        adelta = bdelta
    3 continue
    4 bx = ax - step*dX
      step = step/2.d0
      GOTO 2
    5 RETURN
      END
C
      SUBROUTINE SG (dV,delta)
      implicit real*8 (a-h,o-z)
      real*8 x(6)
      common/const0/pi,tol,iflag,itarget,idim
      common/constopt/alpha,beta,t1,t2
      SELECT CASE (itarget)
        CASE (1)
          CALL initial (dV,x,t)
          CALL TailHooke (X,t,0.d0,delta)
          t1 = t
        CASE (2)
          CALL initial (dV,x,t)
          CALL TailHookm (X,t,0.d0,delta)
          t2= t
      END SELECT
      RETURN
      END
C
      SUBROUTINE HawkeyE (beta1,deltaVf,deltaf)
c     level ONE diagnostics:  target quadrant of SOI
      implicit real*8 (a-h,o-z)
      dimension x(6),xe(6),tj(30,3)
      common/output/tj
      common/constopt/alpha,beta,t1,t2
      common/const0/pi,tol,iflag,itarget,idim
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      itarget = 1              ! Earth
      iflag = 2
      beta = beta1
      deltaV = 0.d0
      t = 0d0
c     >>> first be sure s/c crosses outside of Earth
      cable2 = radE + rpE
      CALL initial (deltaV,x,t)
      CALL TailHooke (X,t,0d0,delta)
      IF (x(1)-xe(1) .LT. cable2) THEN
        dX = +1.d0
      ELSE
        dX = -1.d0
      ENDIF
      IF (x(1) - xe(1) .GT. cable2*1.5d0) GOTO 5
      do 4 i = 1,10
        deltaV = deltaV + 0.005d0*dX
```

```
        CALL initial (deltaV,x,t)
        CALL TailHooke (X,t,0d0,delta)
        IF (x(1) - xe(1) .GT. cable2*1.5d0) GOTO 5
    4 continue
    5 iflag = 1
      delta0 = deltaV
      dX = -1.0*dX
      step = 0.0001d0
      ttol = 1.0d-6
      CALL trapE (delta0,step,dX,ttol,deltaVf)
      deltaVf = delta0
      CALL initial (deltaVf,x,t)
      CALL TailHooke (X,t,0d0,deltaf)
      RETURN
      END
C
      SUBROUTINE earth (x,t,deltaVf)
      implicit real*8 (a-h,o-z)
      real*8 x(6),x2(6),xg(6),x12(6),xe(6),xm(6),xj(6),tj(30,3)
      common/output/tj
      common/const0/pi,tol,iflag,itarget,idim
      common/constopt/alpha,beta,t1,t2
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
      common/constj2/xj,      gmjuptr,rj,omj,phJ
c
      Rf = radE + rpE
      dt = -1000d0
      range1 = helioTOgeo (x,t,3,xg)
      tj(7,1) = xg(1)
      tj(7,2) = xg(2)
      tj(8,1) = xg(4)
      tj(8,2) = xg(5)
      tj(9,1) = x(1)
      tj(9,2) = x(2)
      tj(10,1) = x(4)
      tj(10,2) = x(5)
      tj(6,2) = t/86400d0
      gm = gmsun
      gmsun = gmearth
      gmearth = gm
      SELECT CASE (idim)
        CASE (1)
          phE = pi - (t*ome + phE)
        CASE (2)
          do 2, i = 1,6
    2       xe(i) = -1d0*xe(i)
      END SELECT
      gmmars = 0d0
      gmjupitr = 0d0
      deltaVf = 1000d0
c
      do 75 n=1,250
        CALL RK78T(t,xg,dt,tol,6)
```

```
      range = range2 (xg,x2,x12,sense,Vrel)
      IF (range .LT. radE) THEN
        deltaVf = deltaVf*2d0
        GOTO 78
      ENDIF
      CALL velocity (xg,gmsun,range,Vrel,Rf,deltaV,V1,V2,V3)
      IF (deltaV .LT. deltaVf) THEN
        deltaVf = deltaV
        rangef = range
        xgf = xg(1)
        tj(3,1) = V1
        tj(3,2) = V2
        tj(3,3) = V3
        tj(4,1) = xg(1)
        tj(4,2) = xg(2)
        tj(5,1) = xg(4)
        tj(5,2) = xg(5)
        tj(6,1) = t/86400d0
      ENDIF
      IF (range .GT. soiE) GOTO 78
  75 continue
  78 t1 = t
    CALL constants
    RETURN
    END

    SUBROUTINE TailHooke (x,t,final,deltaV)
    implicit real*8 (a-h,o-z)
    common/output/tj
    common/const0/pi,tol,iflag,itarget,idim
    common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
    real*8 x(6),xe(6),x12(6),tj(30,3)
    dt = -100d0
    delta = soiE*2d0
    do 75 n = 1,1500
      CALL RK78T (t,x,dt,tol,6)
      rhoE = dsqrt(xe(1)**2+xe(2)**2+xe(3)**2)
      rhoX = dsqrt( x(1)**2+ x(2)**2+ x(3)**2)
      cable1 = rhoE + delta
      cable2 = rhoE + soiE
      IF (rhoX .LT. cable1) dt = -86400d0/3d0
      IF (rhoX .LT. cable2) dt = -86400d0/6d0
      SELECT CASE (iflag)
       CASE (1)
        range = range2(x,xe,x12,sense,Vrel)
        IF (sense .GT. 0.d0 .OR. x12(1) .GT. soiE) THEN
          Vrel = Vrel + range/soiE
       CALL velocity (x,gmearth,soiE,Vrel,radE+rpE,deltaV,V1,V2,V3)
          deltaV = deltaV + 0.5d0
          GOTO 78
        ENDIF
        IF (range .LT. soiE) THEN
         IF (final .EQ. 1.d0) THEN
c            CALL Phoenix
```

```
            GOTO 78
          ENDIF
          CALL earth (x,t,deltaV)
          GOTO 78
        ENDIF
      CASE (2)
        alphaX = datan2(x(2),x(1))
        alphaE = datan2(xe(2),xe(1))
        gamma = ABS(alphaX - alphaE)
        IF (gamma .LT. 0.15d0) dt = -4*60*60
        IF (gamma .LT. 0.02d0 .AND. t/86400d0 .LT. -40d0) GOTO 78
     END SELECT
 75 continue
 78 return
    end


c======== sanitized routines =================

    SUBROUTINE velocity (xg,gmsun,range,Vrel,Rf,deltaV,V1,V2,V3)
    implicit real*8 (a-h,o-z)
    dimension xg(6)
    Rp = range
    phi1 = datan(xg(4)/xg(5))
    phi2 = datan(xg(2)/xg(1))
    gamma = ABS(phi1 - phi2)
    Vcirc = dsqrt(gmsun/range)
    V3 =dsqrt( Vrel**2+Vcirc**2-2.d0*Vrel*Vcirc*dcos(gamma))
    V2 = dsqrt(gmsun/Rp)*(1.d0-dsqrt(2.d0/(1.d0+Rp/Rf)))
    V1 = dsqrt(gmsun/Rf)*( dsqrt((2.d0*Rp/Rf)/(1+Rp/Rf))-1.d0)
    deltaV = ABS(V1) + ABS(V2) + ABS(V3)
    RETURN
    END
c
    SUBROUTINE thirdbc(t,n,x3)
    implicit real*8 (a-h,o-z)
    common/const0/pi,tol,iflag,itarget,idim
    common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
    common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
    common/ephemers/ej
    real*8 x3(6),xe(6),xm(6),ej(4,6)
    SELECT CASE (idim)
      CASE (1)
        SELECT CASE (n)
          CASE (3)
            r3 = re
            theta = t*ome + phE
          CASE (4)
            r3 = rm
            theta = t*omm + phM
        END SELECT
        x3(1) = r3*dcos(theta)
        x3(2) = r3*dsin(theta)
        x3(3) = 0.0d0
```

```
  CASE (2)
 SELECT CASE (n)
  CASE (3)
    pe = ej(3,1)*(1d0 - ej(3,2)**2)
    xeM = phE + ome*t
    CALL randv (gmsun,pe,ej(3,2),ej(3,3),ej(3,4),ej(3,5),xeM,xe)
  CASE (4)
    pm = ej(4,1)*(1d0 - ej(3,2)**2)
    xmM = phM + omm*t
    CALL randv (gmsun,pm,ej(4,2),ej(4,3),ej(4,4),ej(4,5),xmM,xm)
  END SELECT
END SELECT
RETURN
END

FUNCTION helioTOgeo (xh,t,n,xg)
implicit real*8 (a-h,o-z)
real*8 xh(6),x3(6),xg(6),xe(6),xm(6)
common/const0/pi,tol,iflag,itarget,idim
common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
SELECT CASE (idim)
  CASE (1)
    SELECT CASE (n)
      CASE (1)
        r3 = rm
        theta = pi/2.d0 + datan(ABS(xh(1))/xh(2))
        x3(1) = r3*dcos(theta)
        x3(2) = r3*dsin(theta)
        x3(3) = 0.0d0
      CASE (3)
        r3 = re
        theta = t*ome + phE
      CASE (4)
        r3 = rm
        theta = t*omm + phM
    END SELECT
    x3(1) = r3*dcos(theta)
    x3(2) = r3*dsin(theta)
    x3(3) = 0.0d0
    iquad = INT ((theta + pi/2.d0)*2.d0/pi)
    SELECT CASE (iquad)
      CASE (1)
        signVx = -1.d0
        signVy = +1.d0
      CASE (2)
        signVx = -1.d0
        signVy = -1.d0
      CASE (3)
        signVx = +1.d0
        signVy = -1.d0
      CASE (4)
        signVx = +1.d0
        signVy = +1.d0
```

```
        END SELECT
        Vcirc = dsqrt(gmsun/r3)
        x3(4) = signVx*ABS(dsin(theta))*Vcirc
        x3(5) = signVy*ABS(dcos(theta))*Vcirc
        x3(6) = 0.d0
        range = range2(xh,x3,xg,sense,Vrel)
      CASE (2)
        SELECT CASE (n)
          CASE (3)
            range = range2(xh,xe,xg,sense,Vrel)
          CASE (4)
            range = range2(xh,xm,xg,sense,Vrel)
        END SELECT
      helioTOgeo = range
      END SELECT
      RETURN
      END
c
      FUNCTION range2 (x1,x2,x12,sense,Vrel)
      implicit real*8 (a-h,o-z)
      common/const0/pi,tol,iflag,itarget,idim
      dimension x1(6),x2(6),x12(6)
c     relative position & velocity of 1 w.r.t. 2
      sense = 1.d0
      range = 0.d0
      Vrel = 0.d0
      rdotV = 0.d0
      do 4 i = 1,3
        j = i+3
        x12(i) = x1(i) - x2(i)
        x12(j) = x1(j) - x2(j)
        range = range + x12(i)**2
        Vrel  = Vrel  + x12(j)**2
        rdotV = rdotV + x12(i)*x12(j)
    4 continue
      range2 = dsqrt(range)
      Vrel   = dsqrt(Vrel)
      beta = dacos(rdotV/(range2*Vrel))
      IF (beta .LT. pi/2) sense = -1.d0
c     range is openinging for beta < 90 deg
      RETURN
      END

      SUBROUTINE constants
      implicit real*8 (a-h,o-z)
      real*8 xe(6),xm(6),xj(6)
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
      common/constj2/xj,      gmjuptr,rj,omj,phJ
      gmsun = 1.32712428d11
      gmearth = 3.986004415d5
      re = 149598023d0                    ! = a
      ome = dsqrt((gmsun+gmearth)/re)/re
      soiE = 924647d0
```

```
      radE = 6378.1363d0
      rpE = 200.d0
      gmmars = 4.305d4
      rm = 227939186d0                    ! = a
      omm = dsqrt((gmsun+gmmars)/rm)/rm
      soiM = 577213d0
      radM = 3397.2d0
      rpM = 80.d0
      gmjuptr = 1.268d8
      rj = 778298361d0                    ! = a
      omj = dsqrt((gmsun+gmjuptr)/rj)/rj
      RETURN
      END
C
      SUBROUTINE Hohmann (x,phase)
      implicit real*8 (a-h,o-z)
      dimension x(6),xe(6),xm(6),ej(4,6),tj(30,3)
      common/output/tj
      common/ephemers/ej
      common/const0/pi,tol,iflag,itarget,idim
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
      CALL ephemerides
      r1 = ej(3,1)
      r2 = ej(4,1)
      a = (r1 + r2)/2.d0
      e = (r2 - r1)/(r1 + r2)
      p = a*(1.d0 - e**2)
      period = pi*(sqrt(a)**3)/dsqrt(gmsun)
      tconj = (pi - period*omm)/(ome-omm)
c
      phase = tconj*ome
      IF (idim .EQ. 2) THEN
        CALL conjunction (t,f0,fe,fm,xMe,xMm)
        phE = xMe
        phM = xMm
        f1 = dmod(fe - phase,2d0*pi)
        f2 = dmod(fm + pi - phase,2d0*pi)
        pe = ej(3,1)*(1d0 - ej(3,2)**2)
        pm = ej(4,1)*(1d0 - ej(4,2)**2)
        r1 = pe/(1d0 + ej(3,2)*dcos(f1))
        r2 = pm/(1d0 + ej(4,2)*dcos(f2))
        a = (r1 + r2)/2.d0
        e = (r2 - r1)/(r1 + r2)
        p = a*(1.d0 - e**2)
        omsc = dsqrt((gmsun)/a)/a
        E0e=2d0*datan(dsqrt((1d0-ej(3,2))/(1d0+ej(3,2)))*tan(f1/2d0))
        e0M= E0e - ej(3,2)*dsin(E0e)
        tconj = (xMe - e0M)/ome
        scM = omsc*tconj
        CALL Ekepler (e,scM,scE)
        scf=2d0*datan(dsqrt((1d0+ej(3,2))/(1d0-ej(3,2)))*tan(scE/2d0))
        omega = f0 - phase
        CALL randv (gmsun,p,e,0.d0,omega,0.d0,scM,x)
```

```
          GOTO 3
        ENDIF
        omsc = dsqrt((gmsun)/a)/a
        scM = omsc*tconj
        omega = 0.d0
      2 CALL randv (gmsun,p,e,0.d0,omega,0.d0,scM,x)
      3 RETURN
        END
C
        SUBROUTINE initial (deltaV,X,t)
        implicit real*8 (a-h,o-z)
        real*8 X(6),xe(6),xm(6),xj(6),tj(30,3)
        common/output/tj
        common/constopt/alpha,beta,t1,t2
        common/const0/pi,tol,iflag,itarget,idim
        common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
        common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
        common/constj2/xj,      gmjuptr,rj,omj,phJ
c
        CALL Hohmann(x,phase)
        IF (idim .EQ. 1) THEN
c            phase = 1.65265383684363d0
           phE = phase
           phM = phase
           phJ = phase  ! pi - dasin(x(2)/rj)
        ENDIF
        IF (beta .NE. 0.d0 .OR. deltaV .NE. 0.d0) THEN
          gamma = ABS(datan(X(5)/X(4)))
          X(4) = X(4) - deltaV*dcos(gamma + beta)
          IF (X(5) .GE. 0.d0) THEN
             X(5) = X(5) + deltaV*dsin(gamma + beta)
          ELSE
             X(5) = X(5) - deltaV*dsin(gamma - beta)
          ENDIF
        ENDIF
        tj(11,1) = x(1)
        tj(11,2) = x(2)
        tj(12,1) = x(4)
        tj(12,2) = x(5)
        t = 0.d0
        RETURN
        END

        SUBROUTINE DERIV(t,x,f)
        implicit real*8 (a-h,o-z)
        real*8 x(7),f(6),xe(6),xm(6),xj(6)
        common/const0/pi,tol,iflag,itarget,idim
        common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
        common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
        common/constj2/xj,      gmjuptr,rj,omj,phJ
        f(1) = x(4)
        f(2) = x(5)
        f(3) = x(6)
        r2 = x(1)**2 + x(2)**2 + x(3)**2
```

```
      r = dsqrt(r2)
      f(4) = -gmsun*(x(1)/r)/r2
      f(5) = -gmsun*(x(2)/r)/r2
      f(6) = -gmsun*(x(3)/r)/r2
      CALL thirdbc(t,3,xe)
      CALL pert3b(gmearth,xe,x,f)
      CALL thirdbc(t,4,xm)
      CALL pert3b(gmmars,xm,x,f)
      IF (idim .EQ. 1) THEN
        CALL thirdbc(t,5,xj)
        CALL pert3b(gmjuptr,xj,x,f)
      ENDIF
      RETURN
      END

      SUBROUTINE ephemerides    ! J2000 (Vallado)
      implicit real*8 (a-h,o-z)
      common/ephemers/ej
      common/const0/pi,tol,iflag,itarget,idim
      real*8 ej (4,6)
      factor = pi/180.d0
c
      ej(3,1) = 149598023.d0           ! a
      ej(3,2) = .016708617d0           ! e
      ej(3,3) = 0.d0                   ! i
      ej(3,4) = 0.d0                   ! LONGITUDE of ascending node
      ej(3,5) = 102.93734808d0*factor   ! longongitude of perihelion
      ej(3,6) = 100.46644851d0*factor   ! true longitude @ epoch
c
      ej(4,1) =    227939186.d0
      ej(4,2) =   .093400620d0
      IF (idim .EQ. 3) THEN
        ej(4,3) =   1.84972648d0*factor
      ELSE
        ej(4,3) = 0.d0
      ENDIF
      ej(4,4) =  49.55809321d0*factor
      ej(4,5) = 336.06023398d0*factor
      ej(4,6) = 355.43327463d0*factor
      RETURN
      END

      SUBROUTINE energye(x,t,E)
      implicit real*8 (a-h,o-z)
      real*8 x(9),xb(6),xe(6)
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      r = dsqrt(x(1)**2 + x(2)**2 + x(3)**2)
      E2body = 0.5d0*(x(4)**2 + x(5)**2 + x(6)**2) - gmsun/r
C
      CALL thirdbc(t,re,ome,phE,xb)
      r23 = dsqrt((xb(1)-x(1))**2 + (xb(2)-x(2))**2 + (xb(3)-x(3))**2)
      Um1 = gmearth/r23
      Um2 = ome*(x(1)*x(5)-x(2)*x(4))
      Um3 = (gmearth/re**3)*(X(1)*xb(1) + X(2)*xb(2))
```

```
C
       E = E2body -Um1 - Um2 + Um3
      RETURN
      END
C
      SUBROUTINE pert3b(GMP,x3,x,f)
      real*8 f(6),x(3),x3(3),gmp,r12,r23
      r12 = dsqrt(x3(1)**2 + x3(2)**2 + x3(3)**2)
      r23 = dsqrt((x3(1)-x(1))**2+(x3(2)-x(2))**2+(x3(3)-x(3))**2)
      f(4) = f(4) - gmp*((x(1)-x3(1))/r23**3 + x3(1)/r12**3)
      f(5) = f(5) - gmp*((x(2)-x3(2))/r23**3 + x3(2)/r12**3)
      f(6) = f(6) - gmp*((x(3)-x3(3))/r23**3 + x3(3)/r12**3)
      RETURN
      END

      SUBROUTINE RK78T(T,X,DT,TOL,N)
      IMPLICIT double precision (A-H,O-Z)
      common/RKCOM/CH(13),AL(13),B(13,12)
      DIMENSION XD(21),F(21,13),X(N),F1(21),F2(21),F3(21),F4(21),F5(21)
c       CALL INRK78 TO INITIALIZE
c      IF(DABS(DT).LT.1.D-20)RETURN
      TM=T
      DT1=DT
C
      DO 40 I=1,N
   40 XD(I)=X(I)
      GO TO 1020
C
 1010 T=TM
      DT1=DT
      DO 41 I=1,N
   41 X(I)=XD(I)
C
 1020 CONTINUE
      CALL DERIV(T,X,F1)
C                        K=2
      DO 602 I=1,N
      TP=B(2,1)*F1(I)
  602 X(I)=XD(I)+DT*TP
      T=TM+AL(2)*DT
      CALL DERIV(T,X,F2)
C                        K=3
      DO 603 I=1,N
      TP=B(3,1)*F1(I)+B(3,2)*F2(I)
  603 X(I)=XD(I)+DT*TP
      T=TM+AL(3)*DT
      CALL DERIV(T,X,F3)
C                        K=4
      DO 604 I=1,N
      TP=B(4,1)*F1(I)+B(4,3)*F3(I)
  604 X(I)=XD(I)+DT*TP
      T=TM+AL(4)*DT
      CALL DERIV(T,X,F4)
C                        K=5
```

```
      DO 605 I=1,N
      TP=B(5,1)*F1(I)+B(5,3)*F3(I)+B(5,4)*F4(I)
  605 X(I)=XD(I)+DT*TP
      T=TM+AL(5)*DT
      CALL DERIV(T,X,F5)
C
      DO 50 K=6,13
      KK=K-1
      DO 71 I=1,N
      TP=B(K,1)*F1(I)+B(K,4)*F4(I)+B(K,5)*F5(I)
      IF(KK.LT.6) GOTO 71
      DO 70 J=6,KK
   70 TP=TP+B(K,J)*F(I,J)
   71 X(I)=XD(I)+DT*TP
      T=TM+AL(K)*DT
   50 CALL DERIV(T,X,F(1,K))
C
      DO 101 I=1,N
      TP=0.D0
C     TP=CH(1)*F1(I)+CH(2)*F2(I)+CH(3)*F3(I)+CH(4)*F4(I)+CH(5)*F5(I)
      DO 100 L=6,13
  100 TP=TP+CH(L)*F(I,L)
  101 X(I)=XD(I)+DT*TP
C
      IF(TOL.EQ.0.D0) GOTO 900
      ER=0.D0
      DO 112 I=1,N
      A=DABS(X(I))
      IF(A.LT.1.D-6) A=1.D-6
      TEI=DABS(F1(I)+F(I,11)-F(I,12)-F(I,13))*CH(12) / A
      IF(TEI.GT.ER) ER=TEI
  112 CONTINUE
      ER=ER*DABS(DT)+1.D-16
      DT=DT*(TOL/ER)**.125D0
      IF(ER.GT.TOL) GOTO 1010
C
  900 CONTINUE
      T=TM+DT1
      RETURN
      END

      SUBROUTINE INRK78
      implicit double precision (a-h,o-z)
      common/RKCOM/CH(13),AL(13),B(13,12)
      DO 1 I=1,13
      CH(I)=0.D0
      AL(I)=0.D0
      DO 1 J=1,12
    1 B(I,J)=0.D0
      CH(6)=34.D0/105.D0
      CH(7)=9.D0/35.D0
      CH(8)=CH(7)
      CH(9)=9.D0/280.D0
      CH(10)=CH(9)
```

```
CH(12)=41.D0/840.D0
CH(13)=CH(12)
AL(2)=2.D0/27.D0
AL(3)=1.D0/9.D0
AL(4)=5.D0/30.D0
AL(5)=5.D0/12.D0
AL(6)=1.D0/2.D0
AL(7)=5.D0/6.D0
AL(8)=5.D0/30.D0
AL(9)=2.D0/3.D0
AL(10)=1.D0/3.D0
AL(11)=1.D0
AL(13)=1.D0
B(2,1)=2.D0/27.D0
B(3,1)=1.D0/36.D0
B(4,1)=5.D0/120.D0
B(5,1)=5.D0/12.D0
B(6,1)=1.D0/20.D0
B(7,1)=-25.D0/108.D0
B(8,1)=31.D0/300.D0
B(9,1)=2.D0
B(10,1)=-91.D0/108.D0
B(11,1)=2383.D0/4100.D0
B(12,1)=3.D0/205.D0
B(13,1)=-1777.D0/4100.D0
B(3,2)=1.D0/12.D0
B(4,3)=1.D0/8.D0
B(5,3)=-25.D0/16.D0
B(5,4)=25.D0/16.D0
B(6,4)=1.D0/4.D0
B(7,4)=125.D0/108.D0
B(9,4)=-53.D0/6.D0
B(10,4)=23.D0/108.D0
B(11,4)=-341.D0/164.D0
B(13,4)=-341.D0/164.D0
B(6,5)=1.D0/5.D0
B(7,5)=-65.D0/27.D0
B(8,5)=61.D0/225.D0
B(9,5)=704.D0/45.D0
B(10,5)=-976.D0/135.D0
B(11,5)=4496.D0/1025.D0
B(13,5)=4496.D0/1025.D0
B(7,6)=125.D0/54.D0
B(8,6)=-2.D0/9.D0
B(9,6)=-107.D0/9.D0
B(10,6)=311.D0/54.D0
B(11,6)=-301.D0/82.D0
B(12,6)=-6.D0/41.D0
B(13,6)=-289.D0/82.D0
B(8,7)=13.D0/900.D0
B(9,7)=67.D0/90.D0
B(10,7)=-19.D0/60.D0
B(11,7)=2133.D0/4100.D0
B(12,7)=-3.D0/205.D0
```

```
      B(13,7)=2193.D0/4100.D0
      B(9,8)=3.D0
      B(10,8)=17.D0/6.D0
      B(11,8)=45.D0/82.D0
      B(12,8)=-3.D0/41.D0
      B(13,8)=51.D0/82.D0
      B(10,9)=-5.D0/60.D0
      B(11,9)=45.D0/164.D0
      B(12,9)=3.D0/41.D0
      B(13,9)=33.D0/164.D0
      B(11,10)=18.D0/41.D0
      B(12,10)=6.D0/41.D0
      B(13,10)=12.D0/41.D0
      B(13,12)=1.D0
      RETURN
      END

*-----------------------------------------------------------

      SUBROUTINE vadd (v,t,n,r)
*     scalar addition of arrays, v(n)+t(n)=r(n)
      real*8 v(n),t(n),r(n)
      DO I=1,n
        r(I)=v(I)+t(I)
      END DO
      RETURN
      END

      SUBROUTINE cross (v,t,r)
*     cross multiply v(3) x t(3) = r(3)
      real*8 v(3),t(3),r(3)
      r(1) = v(2)*t(3) - v(3)*t(2)
      r(2) = v(3)*t(1) - v(1)*t(3)
      r(3) = v(1)*t(2) - v(2)*t(1)
      RETURN
      END

      SUBROUTINE dot (v,w,z)
*     multiply v(3) dot w(3) = z
      real*8 z,v(3), w(3)
      z = v(1)*w(1) + v(2)*w(2) + v(3)*w(3)
      RETURN
      END

      SUBROUTINE ds (s,v,n,r)
*     scalar division of an array with n elements
      real*8 s,v(n),r(n)
      DO I=1,n
            r(I)=v(I)/s
      END DO
      RETURN
      END

      SUBROUTINE exp (v,p,n,r)
```

```
*      raising a vector v(n) to the power p
       real*8 p,v(n), r(n)
       DO I=1,n
              r(I) = v(I)**p
       END DO
       RETURN
       END

       SUBROUTINE  mult (v,t,n,r)
*      vector multiplication of array v(n) with array t(n)
       real*8 v(n),t(n),sum,r
       sum=0.0D0
       DO I=1,n
              sum = sum + v(I)*t(I)
       END DO
       r = sum
       RETURN
       END

       SUBROUTINE  mxm (e,f,n,r)
*      matrix multiplication e(3,3) * f(3,3) = r(3,3)
       implicit real*8 (a-h,o-z)
       real*8 e(3,3), f(3,3), r(3,3)
       DO I=1,n
              sum1=0.0d0
              sum2=0.0d0
              sum3=0.0d0
              DO J=1,n
                     sum1 = sum1 + e(I,J)*f(J,1)
                     sum2 = sum2 + e(I,J)*f(J,2)
                     sum3 = sum3 + e(I,J)*f(J,3)
              END DO
              r(I,1) = sum1
              r(I,2) = sum2
              r(I,3) = sum3
       END DO
       RETURN
       END

       SUBROUTINE norm (v,n,s)
*      calculation of the length of an n-dimensional array v(n)
       real*8 s,v(n),sum
       sum = 0.0d0
       DO I=1,n
              sum = sum + v(I)**2
       END DO
       s = dsqrt(sum)
       RETURN
       END

       SUBROUTINE vr3 (v,m,r)
*      multiplication of 3-element row vector v(3) by matrix m(3,3)
       real*8 v(3),m(3,3),r(3),sum
       sum = 0.0d0
```

```
      DO I=1,3
             DO J=1,3
                    sum = sum + v(I)*m(J,I)
             END DO
             r(I) = sum
             sum = 0.0D0
      END DO
      RETURN
      END

      SUBROUTINE xs (s,v,n,r)
*     scalar multiplication of an array with n elements
      real*8 s,v(n),r(n)
      DO I=1,n
             r(I)=v(I)*s
      END DO
      RETURN
      END

*---------celestial mechanics subroutines-------------------

      SUBROUTINE eccen1 (gmu,r,v,e)
*     calculate eccenticity vector from r,v,mu
      implicit real*8 (a-h,l-z)
      real*8 lrl,lvl,r(3),v(3),rte(3),rvv(3),rvv1(3),ex(3),e(3)
      CALL norm (r,3,lrl)
      CALL norm (v,3,lvl)
      CALL dot (r,v,rv)
      CALL xs (rv,v,3,rvv)
      CALL xs (-1.0D0,rvv,3,rvv1)
      te = lvl**2 - gmu/lrl
      CALL xs (te,r,3,rte)
      CALL vadd (rte,rvv1,3,ex)
      CALL ds (gmu,ex,3,e)
      RETURN
      END

      SUBROUTINE elorb (x,gmu,p,a,lel,i,omega,w,f)
*     calculate orbital elements from position and velocity vectors
      implicit real*8 (a-i,k-z)
      real*8 r(3),v(3),h(3),k(3),n(3),e(3),x(6)
      pi = dacos(-1.d0)
      do 2 j=1,3
       k(j) = 0.d0
       r(j) = x(j)
   2   v(j) = x(j+3)
      CALL norm (r,3,lrl)           ! magnitude of r
      CALL norm (v,3,lvl)           ! magnitude of v
      CALL dot (r,v,z)
      CALL cross (r,v,h)            ! angular momentum vector
      CALL norm (h,3,lhl)           ! magnitude of h
      k(3) = 1.0D0                  ! create [0 0 1] vector
      CALL cross (k,h,n)            ! eccentricity vector
      CALL norm (n,3,lnl)           ! magnitude of n
```

```
      CALL eccen1 (gmu,r,v,e)
      CALL norm (e,3,lel)
      energy = 0.5D0*lvl**2 - gmu/lrl
      a=-0.5D0*gmu/energy
      p=lhl**2/gmu
      i=DACOS(h(3)/dabs(lhl))
      IF (lnl .EQ. 0.d0) THEN
        omega = 0.d0
      ELSE
        omega=DACOS(n(1)/dabs(lnl))
        IF (n(2) .LT. 0.0D0) omega = 2.d0*pi - omega
      ENDIF
      CALL dot (n,e,ne)
      IF (i .eq. 0.d0) THEN
        w = dacos(e(1)/lel)
      ELSE
        w=DACOS(ne/(dabs(lnl*lel)))
      ENDIF
      IF (e(3) .LT. 0.0D0) w = 2.d0*pi - w
      CALL dot (e,r,er)
      f=DACOS(er/(dabs(lel*lrl)))
      IF (z .LT. 0.0D0) f = 2.d0*pi - f
      RETURN
      END

      SUBROUTINE randv (gmu,p,e,i,omega,w,xM,x)
*     calculate position and velocity vectors from orbital elements
      implicit real*8 (a-i,o-z)
      real*8 rx1(3),rx(3),vx(3),vx1(3),ROT(3,3),x(6)
c
      CALL Ekepler (e,xM,xE)
      f = 2.d0*datan(dsqrt((1d0+e)/(1d0-e))*tan(xE/2.d0))
c
      rx1(1)=p*DCOS(f)/(1.0D0+e*DCOS(f))
      rx1(2)=p*DSIN(f)/(1.0D0+e*DCOS(f))
      rx1(3)=0.0D0
      vx1(1)=-1.0D0*DSIN(f)*DSQRT(gmu/p)
      vx1(2)=(e+DCOS(f))*DSQRT(gmu/p)
      vx1(3)=0.0D0
      ROT(1,1)=DCOS(omega)*DCOS(w)-DSIN(omega)*DSIN(w)*DCOS(i)
      ROT(1,2)=-1.0D0*DCOS(omega)*DSIN(w)-DSIN(omega)*DCOS(w)*DCOS(i)
      ROT(1,3)=DSIN(omega)*DSIN(i)
      ROT(2,1)=DSIN(omega)*DCOS(w)+DCOS(omega)*DSIN(w)*COS(i)
      ROT(2,2)=-1.0D0*DSIN(omega)*DSIN(w)+DCOS(omega)*DCOS(w)*DCOS(i)
      ROT(2,3)=-1.0D0*DCOS(omega)*DSIN(i)
      ROT(3,1)=DSIN(w)*DSIN(i)
      ROT(3,2)=DCOS(w)*DSIN(i)
      ROT(3,3)=DCOS(i)
      rx(1)= rx1(1)*ROT(1,1) + rx1(2)*ROT(1,2) + rx1(3)*ROT(1,3)
      rx(2)= rx1(1)*ROT(2,1) + rx1(2)*ROT(2,2) + rx1(3)*ROT(2,3)
      rx(3)= rx1(1)*ROT(3,1) + rx1(2)*ROT(3,2) + rx1(3)*ROT(3,3)
      vx(1)= vx1(1)*ROT(1,1) + vx1(2)*ROT(1,2) + vx1(3)*ROT(1,3)
      vx(2)= vx1(1)*ROT(2,1) + vx1(2)*ROT(2,2) + vx1(3)*ROT(2,3)
      vx(3)= vx1(1)*ROT(3,1) + vx1(2)*ROT(3,2) + vx1(3)*ROT(3,3)
```

```
      do 2 j = 1,3
        x(j)   = rx(j)
2       x(j+3) = vx(j)
      RETURN
      END

      SUBROUTINE  Ekepler (e,aM,aE)
      implicit real*8 (a-h,o-z)
      common/const0/pi,tol,iflag,itarget,idim
      ttol = 0.d-8
      IF (-pi .LT. aM .AND. aM .LT. 0.d0) aE = aM -e
      IF (0.d0 .LT. aM .AND. aM .LT. pi) aE = aM + e
      IF (aM .GT. pi) aE = aM - e
      vary = 100.d0
      do 2 i = 1,50
      bE = aE +(aM - aE + e*dsin(aE))/(1.d0+e*dcos(aE))
      aE = bE
      IF (ABS(aE-bE) .LT. ttol) GOTO 3
2     continue
3     RETURN
      END

      SUBROUTINE conjunction (t,f0e,fe,fm,xMe,xMm)
      implicit real*8 (a-h,o-z)
      common/output/tj
      common/ephemers/ej
      common/constopt/alpha,beta,t1,t2
      common/const0/pi,tol,iflag,itarget,idim
      common/conste2/xe,gmsun,gmearth,radE,re,soiE,ome,phE,rpE
      common/constm2/xm,      gmmars, radM,rm,soiM,omm,phM,rpM
      dimension xe(6),xm(6),ej(4,6),tj(30,3)
c
      pi2 = 2.d0*pi
      xM0e = ej(3,6) - ej(3,4) - ej(3,5)
      xM0m = ej(4,6) - ej(4,4) - ej(4,5)
      period = 1.d0
      t = 0.d0
      test = 2
      do 2 i = 1,1000
        t = t + 86400d0*period
        xMe = xM0e + ome*t
        xMm = xM0m + omm*t
        CALL Ekepler (ej(3,2),xMe,Ee)
        fe=2d0*datan(dsqrt((1d0+ej(3,2))/(1d0-ej(3,2)))*tan(Ee/2d0))
        CALL Ekepler (ej(4,2),xMm,Em)
        fm=2d0*datan(dsqrt((1d0+ej(4,2))/(1d0-ej(4,2)))*tan(Em/2d0))
        f0e = dmod(fe + ej(3,4) + ej(3,5),pi2)
        f0m = dmod(fm + ej(4,4) + ej(4,5),pi2)
        IF (ABS(f0e - f0m) .LT. 1.d0 .AND. f0e .GT. f0m) THEN
          t = t - 86400.d0*period
          period = period/2.d0
          IF (ABS(f0e - f0m) .LT. 1.d-12) GOTO 3
        ENDIF
2     continue
```

```
  3 fe = dmod(fe,pi2)
    xMe = dmod(xMe,pi2)
    pe = ej(3,1)*(1d0 - ej(3,2)**2)
    CALL randv (gmsun,pe,ej(3,2),ej(3,3),ej(3,4),ej(3,5),xMe,xe)
    IF (alpha .NE. 0.d0) THEN
      fm = dmod(fm + alpha,pi2)
      Em=2d0*datan(dsqrt((1d0-ej(4,2))/(1d0+ej(4,2)))*tan(fm/2d0))
      xMm = Em - ej(4,2)*dsin(Em)
      xMm = dmod(xMm,pi2)
    ELSE
      fm = dmod(fm,pi2)
      xMm = dmod(xMm,pi2)
    ENDIF
    pm = ej(4,1)*(1d0 - ej(4,2)**2)
    CALL randv (gmsun,pm,ej(4,2),ej(4,3),ej(4,4),ej(4,5),xMm,xm)
    tj(1,1) = t/86400.d0
    tj(2,1) = f0e*180d0/pi
    tj(2,2) = fe*180d0/pi
    tj(2,3) = fm*180d0/pi
    RETURN
    END

*--------- acknowledgements --------------------------
*
*      The following subroutines were provided by
*      Dr. Rodger Broucke, Professor Emeritus, Univ. of Texas, Austin
*      used with his permission
*
*      INRK78, RK78T, perturb3b, energye, deriv
*
*--------------------------------------------------


    SUBROUTINE gui
    implicit real*8 (a-h,o-z)

    print *, '                Mars Pathfinder '
    print *, '                          by'
    print *, '        Bar X Software - Austin, Texas'
    print *, '                  Est. 1993'
    print *, ' ----------------------------------------------------'
    print *, '.'
    print *, ' Your mission is to reach Mars by adjusting values'
    print *, ' of the trajectory parameters to (1) minimize'
    print *, ' fuel usage, in km/sec of thrust, and (2) to'
    print *, ' reach Mars as quickly as possible. . . . . '
    print *, ' You can change the trajectory in one place, '
    print *, ' about half way to Mars.  You select MAGNITUDE and'
    print *, ' DIRECTION for this thrust.  The program will apply'
    print *, ' the thrust at the best time and place, and'
    print *, ' optimize all the other four maneuvers of the'
    print *, ' flight path based upon your criteria.'
    print *, '.'
    print *, ' Now enter the THRUST MAGNITUDE in km/sec.'
```

```
      print *, ' Values should be between zero'
      print *, ' (a standard Hohmann Transfer) and 2.5.'
      print *, '.'
      read *, dVmars
      print *, '.'
      print *, ' Now the THRUST DIRECTION, in degrees above/below'
      print *, ' the current velocity direction. Be careful!  A value'
      print *, ' below zero, with a low thrust and you will never reach'
      print *, ' Mars because you will pass inside its orbit!'
      print *, '.'
      read *, beta1
      print *, '.'
      print *, ' The program will now compute the minimum energy'
      print *, ' trajectory to get you to Mars in the least time,'
      print *, ' using the values you have specified.'
      print *, ' HEY!'
      print *, ' If the program crashes - dont blame it!!'
      print *, ' Try another scenario - more thrust, 0 angle... '
c
      dVmars = -1.d0*dVmars
      CALL catapult (beta1,beta2,dVmars,deltaVf,tsip)
      call XOUT
c
      print *, '---------------------------------------------------'
      print *, ' Your total thrust requirement in km/sec was'
      print *, deltaVf
      print *, ' The total time required for this mission in days was,'
      print *, tsip
      print *, ' Please refer to the above data printout for a more'
      print *, ' complete mission profile.'
      print *, '---------------------------------------------------'
      print *, '      Copyright 2000, William H. Clark II'
      print *, '             All Rights Reserved.'

      print *, '.'
      print *, '---------------------------------------------------'
      print *, '---------------------------------------------------'
      print *, ' Please enter 1 if you want to try another mission.'
      print *, '.'
      read *, repeat
      IF (repeat .EQ. 1) THEN
    8   print *, ' New thrust,'
        read *, dVmars
        print *, ' New angle,'
        read *, beta1
        print *, '  . . . computing!'
        dVmars = -1.d0*dVmars
        CALL catapult (beta1,beta2,dVmars,deltaVf,tsip)
        call XOUT(LP)
        print *, '.'
        print *, ' Total thrust:'
        print *, deltaVf
        print *, ' Total time:'
        print *, tsip
```

```
      print *, '.'
      print *, ' Enter 1 to try again!'
      print *, '.'
      read *, repeat
      IF (repeat .EQ. 1) THEN
        GOTO 8
      ELSE
        GOTO 9
      ENDIF
    ELSE
      GOTO 10
    ENDIF
  9 close (LP)
 10 RETURN
    END

    SUBROUTINE xout
    implicit real*8 (a-h,o-z)
    dimension tj(30,3)
    common/output/tj
c
    print *, '.'
    print *, '----------------------------------------------------'
    print *, '              Mission Profile'
    print *, '----------------------------------------------------'
    print *, '.'
    print *, ' The program uses the J2000 Ephemerides.  The first'
    print *, ' Earth-Mars conjunction occurs, in days after'
    print *, ' 1/1/2000 (midnight is at 0.5 days) at'
    print *, tj(1,1)
    print *, ' The position of Earth and Mars at this time is'
    print *, ' (units, degrees)'
    print *, tj(2,1)
    print *, ' Where zero degrees is the Vernal Equinox by'
    print *, ' convention.  All subsequent coordinates will be in'
    print *, ' this coordinate system, with the z-axis perpendicular'
    print *, ' to the orbit plane.  The true anomalies of Earth and'
    print *, ' Mars at mid-course correction (about conjunction) are'
    print *, tj(2,2)
    print *, tj(2,3)
    print *, ' The motion is assumed to be co-planar.'
    print *, '----------------------------------------------------'
    print *, '.'
    print *, ' The mission begins with the spacecraft (s/c) in a'
    print *, ' 200 km parking orbit around Earth.  The s/c maneuvers'
    print *, ' to slightly higher circular orbit using a minimum'
    print *, ' energy Hohmann transfer with a thrust of'
    print *, ' (units, km/sec)'
    print *, tj(3,1)
    print *, ' The interplanetary trajectory begins with a thrust of'
    print *, tj(3,3) + tj(3,2)
    print *, ' at the following geocentric (earth-centered)'
    print *, ' coordinates, initiatiating a hyperbolic escape'
    print *, ' trajectory (perturbed by the sun and mars):'
```

```
print *, tj(4,1)
print *, tj(4,2)
print *, ' The initial velocity coordinates are:'
print *, tj(5,1)
print *, tj(5,2)
print *, ' The time (relative to the mid-course thrust) is'
print *, tj(6,1)
print *, ' The s/c leaves Earths sphere of influence (soi)'
print *, ' at the following geocentric coordinates and time:'
print *, tj(7,1)
print *, tj(7,2)
print *, tj(8,1)
print *, tj(8,2)
print *, tj(6,2)
print *, ' Its heliocentric (sun-centered) coordinates'
print *, ' at this time are'
print *, tj(9,1)
print *, tj(9,2)
print *, tj(10,1)
print *, tj(10,2)
print *, '---------------------------------------------------'
print *, '.'
print *, ' The s/c is now on an elliptical heliocentric course'
print *, ' (perturbed by Earth, Mars, and Jupiter).  The mid-'
print *, ' course correction occurs at time zero, and at the'
print *, ' following heliocentric coordinates'
print *, tj(11,1)
print *, tj(11,2)
print *, tj(12,1)
print *, tj(12,2)
print *, ' (This velocity is after the thrust is applied.)'
print *, ' The s/c reaches Mars soi at the heliocentric'
print *, ' coordinates and time of,'
print *, tj(13,1)
print *, tj(13,2)
print *, tj(14,1)
print *, tj(14,2)
print *, tj(15,1)
print *, ' (Time, measured after the mid-course corrrection.)'
print *, ' The mars-centered (geocentric) coordinates at this'
print *, ' point are'
print *, tj(16,1)
print *, tj(16,2)
print *, tj(17,1)
print *, tj(17,2)
print *, '---------------------------------------------------'
print *, '.'
print *, ' The s/c follows a hyperbolic capture orbit to Mars'
print *, ' (perturbed by the sun), and applies a thrust of'
print *, tj(21,3) + tj(21,1)
print *, ' to circularize the orbit at coordinates, and time,'
print *, tj(18,1)
print *, tj(18,2)
print *, tj(19,1)
```

```
print *, tj(19,2)
print *, tj(20,1)
print *, ' A final thrust, used to target a specific latitude'
print *, ' and longitude on the surface of Mars, of'
print *, tj(21,2)
print *, ' brings the s/c via Hohmann Transfer into a 100 km'
print *, ' parking orbit around Mars.  The total time and'
print *, ' thrust required for this scenario are'
print *, tj(22,1)
print *, tj(22,2)
print *,'.'
RETURN
END
```