The Journal of the Astronautical Sciences, Vol. 55, No 3, July–September 2007, pp. 349–371

Optimal Lunar Swingby Trajectories[1]

John T. Betts[2]

Background and Motivation

With the growth of the space age, it has become desirable to place spacecraft into a wide variety of mission orbits. Typically a launch vehicle is used to deploy a satellite into a low altitude park orbit. It then becomes necessary to transfer from the park orbit to the mission orbit. A major goal of mission design is to compute this *orbit transfer* to minimize some performance objective. The most common objective is to minimize the fuel consumed by the orbit transfer. For spacecraft using propulsion systems whose thrust is high compared to the mass of the vehicle, i.e. so called *high thrust* systems, the duration of the burns is very short compared to the duration of the orbit transfer itself. Consequently it is common to model high thrust systems using an instantaneous or *impulsive* velocity change. In essence a minimum fuel orbit transfer is approximated using a so-called *minimum* Δv *transfer.* In 1925, the German scientist, Walter Hohmann, demonstrated that a minimum Δv transfer between two circular orbits in the same plane could be achieved using two impulsive velocity increments.

Although the original *Hohmann transfer* was strictly between coplanar orbits, the term is often loosely used to describe any two-impulse transfer. For example, one of the most common applications involves transferring between a circular orbit with an altitude of 150 nm and inclination of 28.5 deg (a standard shuttle park orbit) to a geosynchronous orbit which is circular at an altitude of 19,323 nm and inclination of 0.0 deg. The "Hohmann" transfer for this application is illustrated in Fig. 1. However the park and mission orbits are not coplanar in this case, and some of the requisite inclination change must be accomplished by each of the Δv. For this transfer nearly all of the inclination change ($\approx$26 deg) is achieved by the second impulse. In fact for any orbit transfer it is most efficient to change the orbital plane at a high altitude when the velocity is smallest.

[1]Presented as an invited "Senior Lecture on Trajectory Optimization" at the 3rd International Workshop on Astrodynamic Tools and Techniques, Oct. 2–5, 2006, ESTEC, Noordwijk, The Netherlands.

[2]Mathematics and Engineering Analysis, The Boeing Company, P.O. Box 3707, MS 7L-21, Seattle, Washington 98124-2207.

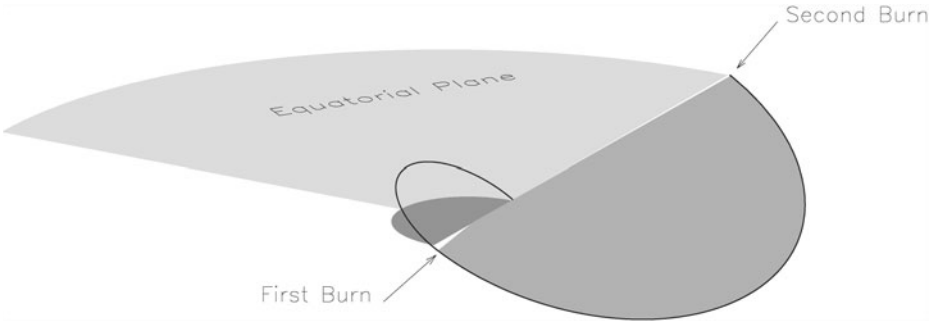


FIG. 1. Optimal Two-Impulse Transfer to Geosynchronous Orbit.

Because changing the orbit plane is the most expensive portion of the transfer one is led to consider a *three burn transfer*. For this type of transfer, the second burn does all of the plane change at a very high altitude. In fact it was demonstrated in Betts [2] that a three impulse solution is more efficient than two impulses for some transfers that require large plane changes. Figure 2 illustrates the situation. A critical property of a three burn transfer is that the second burn be located a "long" distance from the primary body (Earth). Thus one is led to consider locating the second impulse at or near the Moon. In effect the lunar gravitational attraction can be used in lieu of the second burn, thereby producing a very efficient trajectory. The concept of using a lunar gravity assist to design a nominal mission trajectory has been considered by many authors (cf [16]). Lee, Dunham, Hsu, and Roberts [13] discuss using a lunar swingby, however as with most analyses they do not minimize the impulsive velocity.

This paper deals with an approach for computing optimal lunar swingby trajectories between two Earth orbits. First some representative optimal transfers are presented that illustrate the performance benefits. Then the problem formulation is explained and a number of challenging numerical issues are addressed. To facilitate using the examples as benchmark problems, the dynamics are modeled using three-body mechanics but do not require more sophisticated planetary ephemerides. Finally, suggestions are given concerning the numerical methods as well as practical implementation of the technique.

Optimal Lunar Transfer Examples

For the sake of illustration, examples of lunar swingby transfers to four different mission orbits are presented. All of the transfers begin in a circular park orbit at an

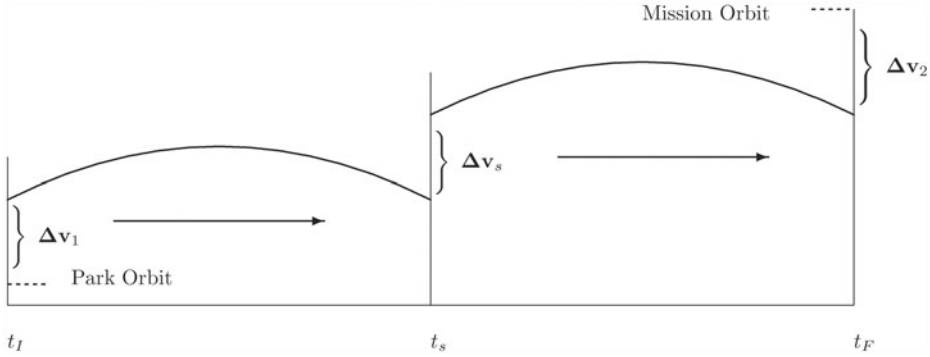


FIG. 2. Optimal Three-Impulse Transfer Orbit.

altitude of 150 nm, with an inclination of 28 deg. The outbound transfer orbit is established by $\mathbf{\Delta v}_1$, and after passing around the Moon the inbound transfer returns to the mission orbit which is established by applying $\mathbf{\Delta v}_2$. All of the transfers minimize the total Δv, that is

$$F = \|\mathbf{\Delta v}_1\| + \|\mathbf{\Delta v}_2\| \tag{1}$$

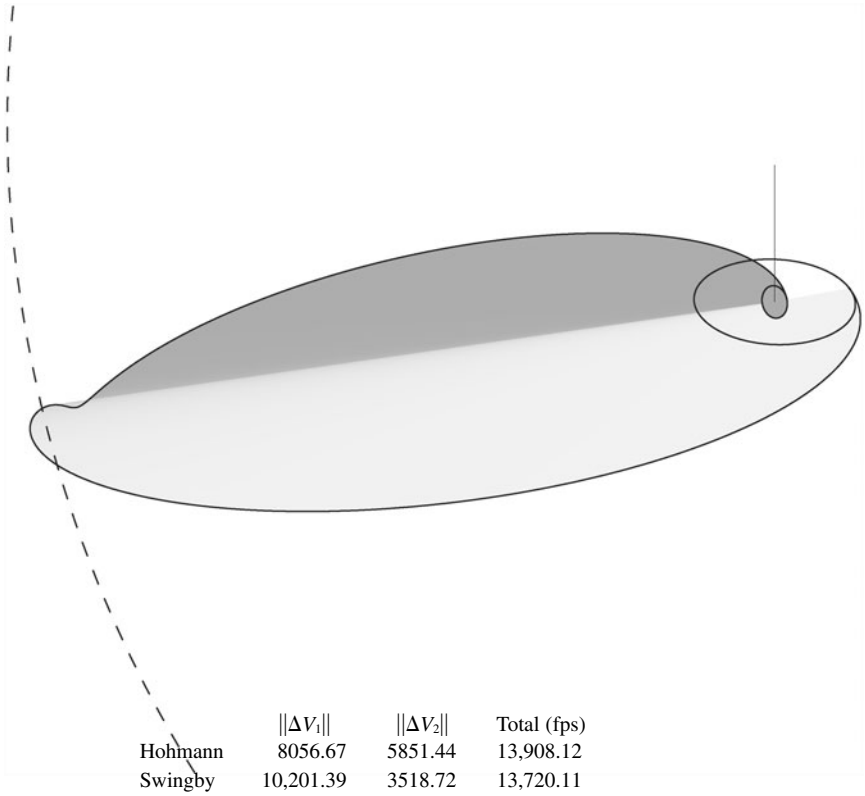
To obtain meaningful results it is also necessary to impose some limit on the total transfer time. An upper bound of 15 days is imposed on the total transfer time for the examples.

Synchronous Equatorial

Results are presented for a geosynchronous orbit which is circular at an altitude of 19,323 nm and inclination of 0.0 deg. Figure 3 illustrates the optimal lunar swingby trajectory and for comparison summarizes the total Δv of the swingby as well as a standard two-impulse "Hohmann" transfer. It is interesting to observe that even for this case which requires only 28.5 degrees of plane change, the swingby solution saves 188.01 fps of total Δv.

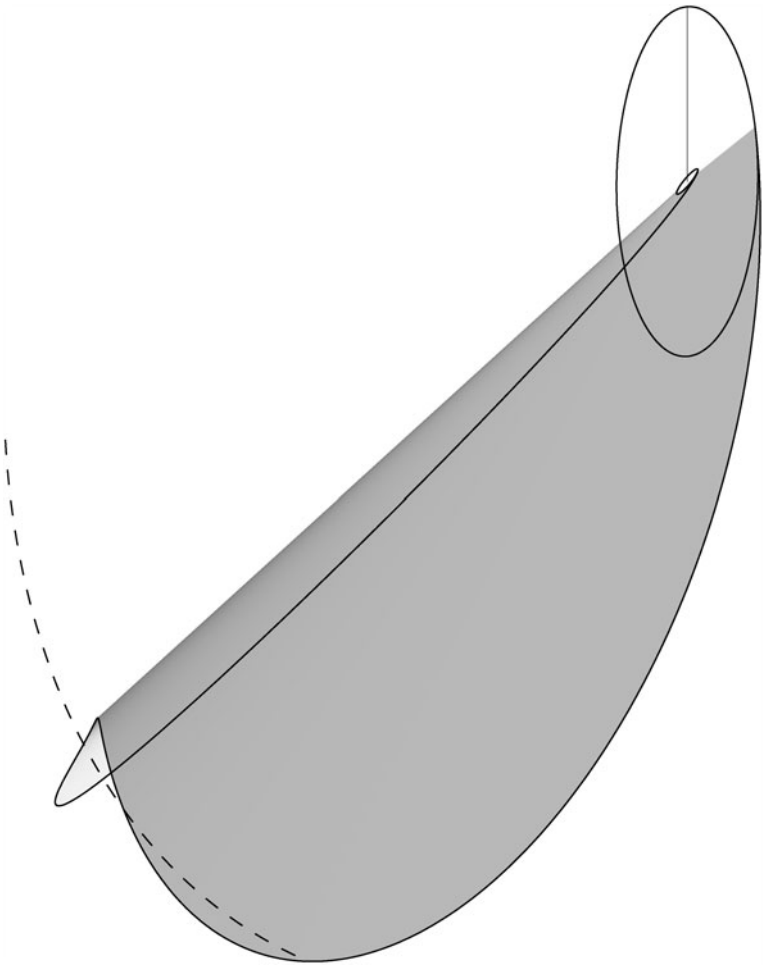
Polar, 24 hr (A)

Figure 4 illustrates the optimal lunar swingby trajectory to an orbit which is circular at an altitude of 19,323 nm and inclination of 90 deg. This solution is characterized by an outbound transfer trajectory from the descending node of the park



	$\|\Delta V_1\|$	$\|\Delta V_2\|$	Total (fps)
Hohmann	8056.67	5851.44	13,908.12
Swingby	10,201.39	3518.72	13,720.11

FIG. 3. Optimal Swingby Transfer to Geosynchronous Orbit.



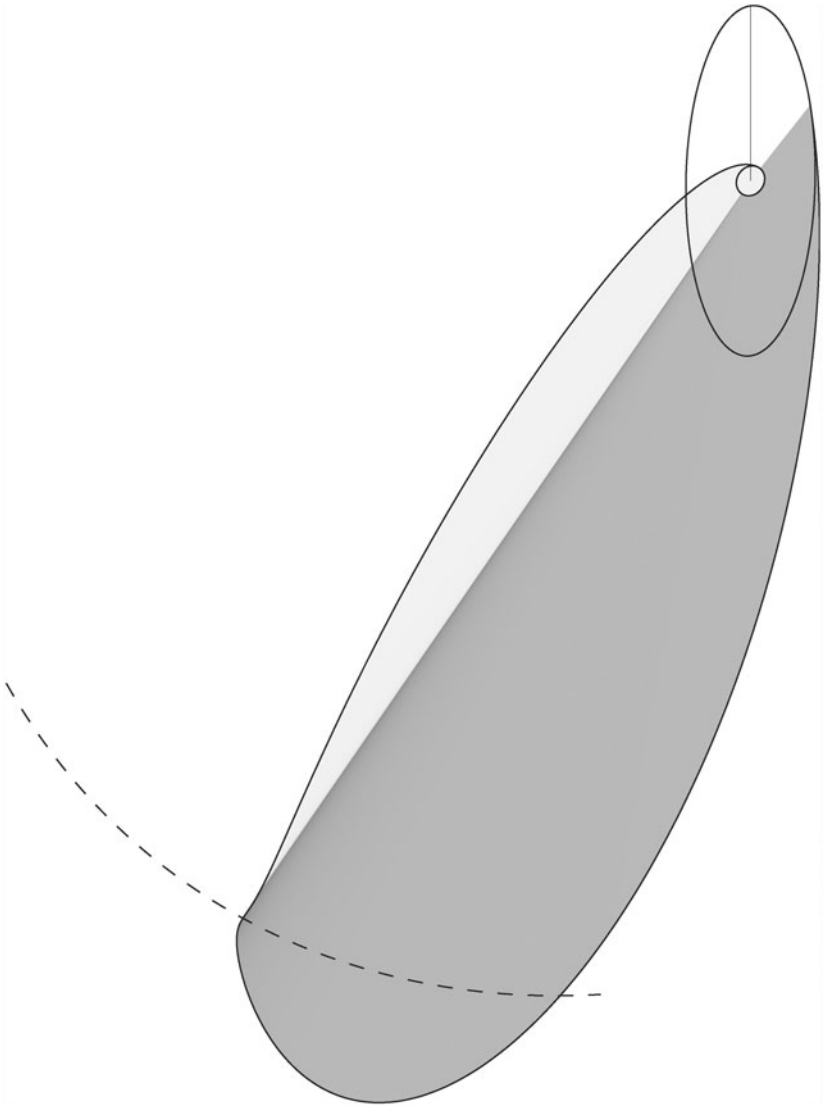
	$\|\Delta V_1\|$	$\|\Delta V_2\|$	Total (fps)
Hohmann	8113.34	8610.82	16,724.17
Swingby	10,225.16	3440.44	13,665.61

FIG. 4. Optimal Swingby Transfer to Polar, 24 hr Orbit.

orbit and as such will be referred to as solution "A." The total plane change for this transfer is 61.5 deg which is achieved using 3059.56 fps less than the corresponding Hohmann transfer.

Polar, 24 hr (B)

Figure 5 illustrates the optimal lunar swingby trajectory to the same mission orbit as in the previous section. In contrast to solution "A," this trajectory is characterized by an outbound transfer from the ascending node of the park orbit and as such will be referred to as solution "B." Although the swingby solution "B" is slightly less efficient than solution "A" it still is 3013.20 fps less than the Hohmann transfer. It is also worth noting the very small difference in the Hohmann solutions between "A" and "B," which can be attributed to the small (yet different) lunar perturbations.

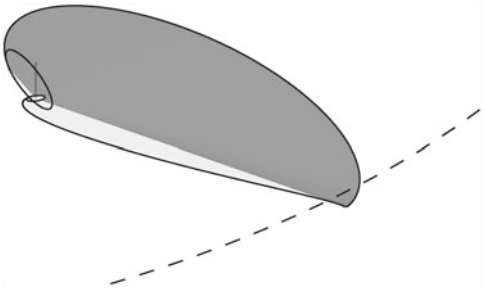


	$\|\|\Delta V_1\|\|$	$\|\|\Delta V_2\|\|$	Total (fps)
Hohmann	8113.33	8610.77	16,724.10
Swingby	10,251.24	3459.66	13,710.90

FIG. 5. Optimal Swingby Transfer to Polar, 24 hr Orbit.

Retrograde Molniya

The final example, shown in Fig. 6, is a retrograde Molniya mission orbit. Specifically the orbit is elliptical with an (osculating) apogee altitude of 21,450 nm, a perigee altitude of 350 nm, an argument of perigee of 270 deg, and a retrograde inclination of 116.6 deg. This particular orbit has a period of approximately 12 hours, and requires an orbital plane change of 88.1 deg. For this example the lunar swingby saves 25,011.05 fps of total Δv. In this extreme case the Hohmann transfer is approximately 264% more expensive.



	$\|\Delta V_1\|$	$\|\Delta V_2\|$	Total (fps)
Hohmann	546.99	39,682.76	40,229.76
Swingby	10,392.39	4826.32	15,218.71

FIG. 6. Optimal Swingby Transfer to Molniya Orbit.

Equations of Motion

The dynamic behavior of a spacecraft can be modeled using a simplified version of the N-body problem as described in Battin [1]. It is convenient to use an Earth-centered Cartesian coordinate system, with boldface notation used to distinguish a vector from a scalar. Let $\mathbf{r}$ and $\mathbf{v}$ denote the position and velocity of the spacecraft, and $\mathbf{r}_L$ and $\mathbf{v}_L$ the position and velocity of the Moon.[3] The motion is described by

$$\dot{\mathbf{r}} = \mathbf{v} \tag{2}$$

$$\dot{\mathbf{v}} = -\frac{\mu_e}{r^3}\,\mathbf{r} + \mathbf{g}_L \tag{3}$$

$$\dot{\mathbf{r}}_L = \mathbf{v}_L \tag{4}$$

$$\dot{\mathbf{v}}_L = -\frac{\mu_o}{r_L^3}\,\mathbf{r}_L \tag{5}$$

where $\|\mathbf{r}\| = r$, $\|\mathbf{r}_L\| = r_L$, and $\mu_o = \mu_e + \mu_L = Gm_e + Gm_L$ where G is the universal gravitation constant with m_e and m_L denoting the masses of the Earth and Moon respectively. The disturbing acceleration caused by the gravitation of the Moon is

$$\mathbf{g}_L = -\mu_L\left[\frac{1}{d^3}\,\mathbf{d} + \frac{1}{r_L^3}\,\mathbf{r}_L\right] \tag{6}$$

where

$$\mathbf{d} = \mathbf{r} - \mathbf{r}_L \tag{7}$$

is a vector from the Moon to the vehicle with magnitude $\|\mathbf{d}\| = d$. To avoid losing precision when evaluating equation (6), Battin [1] suggests defining the function

$$F(q) = q\left[\frac{3 + 3q + q^2}{1 + \left(\sqrt{1+q}\right)^3}\right] \tag{8}$$

where

$$q = \frac{\mathbf{r}^{\mathsf{T}}(\mathbf{r} - 2\mathbf{r}_L)}{\mathbf{r}_L^{\mathsf{T}}\mathbf{r}_L} \tag{9}$$

[3]Subscript Convention: x_p park; x_m mission; x_L lunar; x_i inbound; x_o outbound; x_s swingby.

The perturbing acceleration of equation (6) is then given by

$$\mathbf{g}_L = -\frac{\mu_L}{d^3}[\mathbf{r} + F(q)\mathbf{r}_L] \tag{10}$$

Kepler Orbit Propagation

As written the differential equations (4–5) constitute a *two-body* approximation to the lunar motion. It is well known that this system has an analytic solution and for convenience we briefly summarize the approach (cf [1], [9]). In general, the technique is applicable to any two-body system using the appropriate definitions for the gravitational constant μ_o and coordinates $(\mathbf{r}, \mathbf{v})$ and consequently we omit the subscript for generality. Given a Cartesian state vector $\mathbf{r}_o$, $\mathbf{v}_o$ at time t_o called the *reference epoch* and a specified change in eccentric anomaly $\Delta E = E - E_o$, a new state vector $(\mathbf{r}, \mathbf{v})$, with corresponding time change $\Delta t = t - t_o$ can be computed as

$$r_o = \|\mathbf{r}_o\| \tag{11}$$

$$\sigma_o = \frac{1}{\sqrt{\mu_o}}\mathbf{r}_o^T\mathbf{v}_o \tag{12}$$

$$v_o^2 = \mathbf{v}_o^T\mathbf{v}_o \tag{13}$$

$$\frac{1}{a} = \frac{2}{r_o} - \left[\frac{v_o^2}{\mu_o}\right] \tag{14}$$

$$\rho = 1 - \frac{r_o}{a} \tag{15}$$

$$C = a(1 - \cos\Delta E) \tag{16}$$

$$S = \sqrt{a}\sin\Delta E \tag{17}$$

$$F = 1 - \frac{C}{r_o} \tag{18}$$

$$G = \frac{1}{\sqrt{\mu_o}}(r_o S + \sigma_o C) \tag{19}$$

$$r = r_o + \rho C + \sigma_o S \tag{20}$$

$$F_t = -\frac{\sqrt{\mu_o}}{r r_o}S \tag{21}$$

$$G_t = 1 - \frac{C}{r} \tag{22}$$

$$\mathbf{r} = F\mathbf{r}_o + G\mathbf{v}_o \tag{23}$$

$$\mathbf{v} = F_t\mathbf{r}_o + G_t\mathbf{v}_o \tag{24}$$

$$\Delta t = \sqrt{\frac{a^3}{\mu_o}}\left[\Delta E + \frac{\sigma_o C}{a\sqrt{a}} - \rho\frac{S}{\sqrt{a}}\right] \tag{25}$$

Notice that the sequence of calculations equations (11–25) constitute an *explicit* definition for the states, as well as the corresponding time increment, that can be expressed as

$$\mathbf{r} = \mathbf{h}_r(\Delta E) \tag{26}$$

$$\mathbf{v} = \mathbf{h}_v(\Delta E) \tag{27}$$

$$\Delta t = h_t(\Delta E) \tag{28}$$

The propagation is explicit with respect to the independent variable ΔE, but is *implicit* with respect to the time t. To be more precise let us rewrite equation (28) as

$$t_E = t_o + h_t(\Delta E) \tag{29}$$

since $\Delta t = t_E - t_o$ where t_E is the final time corresponding to the eccentric anomaly increment ΔE.

Differential-Algebraic Formulation of Three Body Dynamics

The original system of 12 differential equations (2–5) can be recast as a differential-algebraic (DAE) system involving the six states $\mathbf{r}(t)$ and $\mathbf{v}(t)$, and the single algebraic variable $\Delta E(t)$ as

$$\dot{\mathbf{r}} = \mathbf{v} \tag{30}$$

$$\dot{\mathbf{v}} = -\frac{\mu_e}{r^3}\mathbf{r} + \mathbf{g}_L \tag{31}$$

$$0 = t - t_o - h_t(\Delta E) \tag{32}$$

Observe that the implicit algebraic equation (32) is a modified form of Kepler's transcendental equation. The gravitational perturbation on the spacecraft $\mathbf{g}_L$ is defined by the position vectors $\mathbf{r}$ and $\mathbf{r}_L$ using equations (8–10). The lunar position vector is completely determined by equation (26) using the Kepler propagation procedure outlined in equations (11–25). Thus the complete functional dependency is given by

$$\mathbf{g}_L = \mathbf{g}_L(\mathbf{r}, \Delta E)$$

Boundary Conditions

Denote the state variables at a particular time (e.g. the beginning or end of the trajectory) by $\tilde{\mathbf{r}}$ and $\tilde{\mathbf{v}}$. Boundary conditions defining both the park and mission orbits are typically stated in terms of *osculating* orbit elements, i.e. nonlinear functions of $\tilde{\mathbf{r}}$ and $\tilde{\mathbf{v}}$. There are many equivalent ways to specify both the park and mission orbits. A circular orbit with specified radius $\bar{r}$ can be achieved by imposing the boundary constraints

$$\bar{r} = \|\tilde{\mathbf{r}}\| \tag{33a}$$

$$\sqrt{\frac{\mu_e}{\bar{r}}} = \|\tilde{\mathbf{v}}\| \tag{33b}$$

$$0 = \frac{\tilde{\mathbf{r}}^{\mathsf{T}}\tilde{\mathbf{v}}}{\sqrt{\mu_e \bar{r}}} \tag{33c}$$

Equation (33a) fixes the magnitude of the position vector, equation (33b) defines the circular velocity, and equation (33c) ensures the flight path angle (and eccentricity) is zero. Observe that when equations (33a) and (33b) are satisfied equation (33c) is just

$$\frac{\tilde{\mathbf{r}}^{\mathsf{T}}\tilde{\mathbf{v}}}{\sqrt{\mu_e \bar{r}}} = \frac{\tilde{\mathbf{r}}^{\mathsf{T}}\tilde{\mathbf{v}}}{\|\tilde{\mathbf{r}}\|\|\tilde{\mathbf{v}}\|}$$

which is equivalent to making the normalized position and velocity vectors orthogonal. However, as written the denominator in equation (33c) is a *constant,* which is the preferable, i.e. "more linear," way to pose the constraint.

To achieve a particular inclination $\bar{i}$ we must enforce the boundary condition

$$\cos \bar{i} = \hat{\mathbf{h}}_3 \tag{34a}$$

where the angular momentum vector $\mathbf{h}$ and north vector $\mathbf{i}_z$ are

$$\mathbf{h} = \tilde{\mathbf{r}} \times \tilde{\mathbf{v}} \tag{34b}$$

$$\hat{\mathbf{h}} = \frac{\mathbf{h}}{\|\mathbf{h}\|} \tag{34c}$$

$$\mathbf{i}_z = (0, 0, 1)^{\mathsf{T}} \tag{34d}$$

$$\hat{\mathbf{h}}_3 = \mathbf{i}_z^{\mathsf{T}}\hat{\mathbf{h}} \tag{34e}$$

Observe that equation (34a) is stated in terms of $\cos \bar{i}$ rather than the inclination $\bar{i}$ itself, thereby avoiding ambiguities when computing the inverse cosine.

If the desired orbit is elliptic, equations (33a)–(33c) are not applicable. Instead to achieve a specified semimajor axis $\bar{a}$, and eccentricity $\bar{e}(> 0)$, we must impose the boundary conditions

$$\bar{a} = \left(\frac{2}{\|\tilde{\mathbf{r}}\|} - \frac{\|\tilde{\mathbf{v}}\|^2}{\mu_e} \right)^{-1} \tag{35a}$$

$$\bar{e} = \|\mathbf{e}\| \tag{35b}$$

where the eccentricity vector is given by

$$\mathbf{e} = \frac{1}{\mu_e}(\tilde{\mathbf{v}} \times \mathbf{h}) - \frac{\tilde{\mathbf{r}}}{\|\tilde{\mathbf{r}}\|} \tag{35c}$$

and the angular momentum is given by equation (34b).

For elliptic orbits ($\|\mathbf{e}\| > 0$), the eccentricity vector is directed toward periapsis, thereby defining the orientation of the principal axis. In general, the argument of periapsis ω is an angle measured in the orbital plane from the ascending node to periapsis. The particular case $\bar{\omega} = 270°$, can be enforced by imposing the conditions

$$0 = \hat{\mathbf{e}}^{\mathsf{T}}\mathbf{i}_n \tag{36a}$$

$$0 > \hat{\mathbf{e}}_3 \tag{36b}$$

where

$$\hat{\mathbf{e}} = \frac{\mathbf{e}}{\bar{e}} \tag{36c}$$

$$\hat{\mathbf{e}}_3 = \mathbf{i}_z^{\mathsf{T}}\hat{\mathbf{e}} \tag{36d}$$

$$\mathbf{i}_n = \mathbf{i}_z + \hat{\mathbf{h}} \tag{36e}$$

and $\mathbf{e}$ is given by equation (35c), with $\hat{\mathbf{h}}$ defined by equation (34b), and $\mathbf{i}_z$ from equation (34d). Again notice that the denominator of equation (36c) involves the *constant* $\bar{e}$ instead of the quantity $\|\mathbf{e}\|$ appearing on the right hand side of equation (35b).

A Four Step Solution Technique

Most effective numerical methods are based on the premise that:

A hard problem can be broken into a sequence of easy subproblems.

For example, to compute the root of a *nonlinear* constraint $c(x) = 0$ using Newton's Method, one must solve the sequence of *linear* approximations $x_{k+1} = x_k - c(x_k)/c'(x_k)$. Similarly, when using a nonlinear programming (NLP) algorithm (cf [3]) to solve a *nonlinear optimization* problem, one solves a sequence of: (a) *quadratic programming* subproblems (an SQP Method) or, (b) *unconstrained* subproblems (a barrier method). In like fashion an optimal control solution can be obtained by solving a sequence of NLP subproblems. Of course there may be no clear definition of an "easy subproblem." For Newton's method is it better to compute $c'(x_k)$ or use a secant approximation? Is a quadratic program easier than an unconstrained subproblem? Ultimately the solution technique should consist of a sequence of subproblems that can be solved efficiently and reliably. Thus to compute a solution to an *optimal lunar swingby* let us first pose a sequence of *"easier"* subproblems. It is important to remark that the proposed technique is a heuristic—other equally valid methods can be postulated. However the heuristic has proven to be both reliable and efficient. A later section describes *how* each subproblem is solved.

The *Four Step Solution Technique* can be summarized as follows:

Step 1: Three Impulse, Conic Solution
Solve a small NLP with analytic propagation ignoring lunar gravity.
Step 2: Three-Body Approximation to Conic Solution
Solve an "inverse problem" to fit three-body dynamics to the conic solution.
Step 3: Optimal Three-Body Solution with Fixed Swingby Time
Use the solution from Step 2 to initialize.
Step 4: Optimal Three-Body Solution
Compute solution with free swingby time, using Step 3 as an initial guess.

Step 1: Three Impulse, Conic Solution

Since the optimal trajectory depends on the relative geometry of three bodies, namely, the Earth, Moon, and spacecraft, it is critical that the solution process is initiated with a reasonable geometric configuration. This goal can be achieved by using a very simplified model of the dynamics. In particular for Step 1, it is assumed that the spacecraft dynamics are determined entirely by the gravitational attraction of the primary body, Earth. Further, it is assumed that the lunar swingby can be approximated by a simple velocity increment. Because all of the dynamics are modeled using a two-body approximation, the analytic trajectory propagation approach outlined in the Kepler Orbit Propagation section can be exploited. With these simplifying assumptions one can pose the following very simple nonlinear programming problem.

Optimization Variables. The problem can be formulated using 24 variables to define the trajectory as illustrated in Fig. 7.

$$(\mathbf{r}_o, \mathbf{v}_o, \Delta\mathbf{v}_1, \Delta E_o): \quad \text{State at Park Orbit Departure} \tag{37}$$

$$(\mathbf{r}_i, \mathbf{v}_i, \Delta\mathbf{v}_2, \Delta E_i): \quad \text{State at Mission Orbit Arrival} \tag{38}$$

$$(\Delta\mathbf{v}_s, \Delta E_L): \quad \text{Swingby Velocity Increment and Lunar Transfer Angle to Intercept} \tag{39}$$

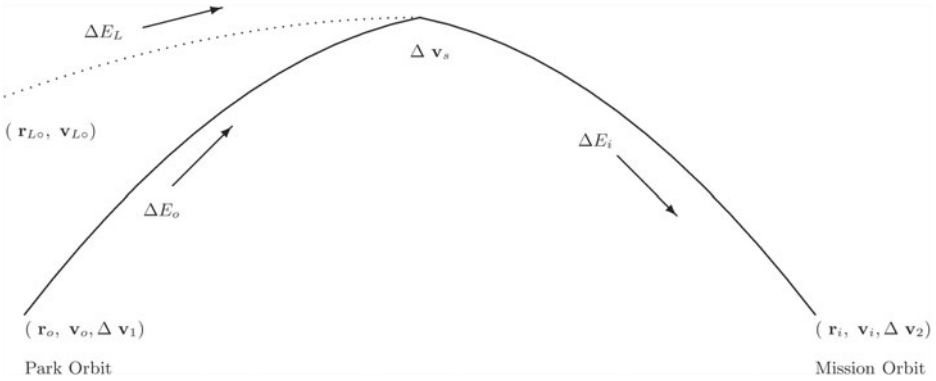


FIG. 7. Three Impulse Conic Solution.

The outbound (Earth to Moon) transfer orbit is defined by the variables in equation (37), namely the Cartesian state at the beginning of the outbound transfer $\mathbf{r}_o$, $\mathbf{v}_o$, the impulsive velocity $\Delta\mathbf{v}_1$, and the eccentric anomaly change for the outbound trajectory ΔE_o. Equation (38) defines the corresponding variables for the inbound (Moon to Earth) transfer. The velocity increment provided by the Moon at swingby is defined as $\Delta\mathbf{v}_s$, and the location of the Moon relative to a reference epoch is defined by the angle ΔE_L.

Park Orbit Conditions. In order to enforce the boundary conditions at the park orbit, the following NLP constraints must be imposed:

$$\mathbf{r}_p = \mathbf{r}_o \qquad \text{Position Continuity} \tag{40}$$

$$\mathbf{v}_p = \mathbf{v}_o - \Delta\mathbf{v}_1 \qquad \text{Impulsive Velocity Change} \tag{41}$$

$$\boldsymbol{\phi}_p(\mathbf{r}_p, \mathbf{v}_p) = \mathbf{0} \qquad \text{Park Orbit Constraints} \tag{42}$$

Equation (40) insures that the park orbit and outbound transfer orbit position is continuous. The impulsive velocity change at departure is enforced by equation (41). The park orbit state vector must also satisfy the nonlinear constraints denoted $\boldsymbol{\phi}_p$ in equation (42). For the circular park orbit illustrated here, the vector $\boldsymbol{\phi}_p$ is given by equations (33a–c) and equation (34a).

Mission Orbit Conditions. The boundary conditions imposed by the mission orbit lead to a set of NLP constraints similar to those enforced at departure, namely,

$$\mathbf{r}_m = \mathbf{r}_i \qquad \text{Position Continuity} \tag{43}$$

$$\mathbf{v}_m = \mathbf{v}_i + \Delta\mathbf{v}_2 \qquad \text{Impulsive Velocity Change} \tag{44}$$

$$\boldsymbol{\phi}_m(\mathbf{r}_m, \mathbf{v}_m) = \mathbf{0} \qquad \text{Mission Orbit Constraints} \tag{45}$$

Constraints (43) and (44) are analogous to (40) and (41). The mission orbit constraints denoted by the vector $\boldsymbol{\phi}_m$ in equation (45), are computed using the appropriate expressions from the sixth section.

Lunar Conditions. The conditions at the Moon used to approximate the lunar swingby all involve expressions for the state vector computed using the Kepler propagation described by equations (26) and (27) as

$$\mathbf{h}_r(\mathbf{r}_o, \mathbf{v}_o, \Delta E_o) = \mathbf{h}_r(\mathbf{r}_i, \mathbf{v}_i, \Delta E_i) \quad \text{Outbound/Inbound Position} \quad (46)$$

$$\mathbf{h}_r(\mathbf{r}_o, \mathbf{v}_o, \Delta E_o) = \mathbf{h}_r(\mathbf{r}_{Lo}, \mathbf{v}_{Lo}, \Delta E_L) \quad \text{Outbound/Lunar Position} \quad (47)$$

$$\mathbf{h}_v(\mathbf{r}_i, \mathbf{v}_i, \Delta E_i) = \mathbf{h}_v(\mathbf{r}_o, \mathbf{v}_o, \Delta E_o) + \mathbf{h}_v(\mathbf{r}_{Lo}, \mathbf{v}_{Lo}, \Delta E_L) + \Delta \mathbf{v}_s \quad \text{Velocity Change} \quad (48)$$

The constraints (46) force the outbound and inbound transfer trajectories to have the same position at the swingby. Observe that the outbound position $\mathbf{h}_r$ is completely determined by the departure state $\mathbf{r}_o$, $\mathbf{v}_o$, and the outbound eccentric anomaly change ΔE_o, with similar comments applicable to the inbound leg. While equation (46) insures continuity between the outbound and inbound trajectories, it is also necessary that the swingby occur at the Moon's position. This is achieved using constraint (47). Of course this constraint will locate the swingby at the *center* of the Moon, which can only be achieved by ignoring the lunar gravity. Finally, one must model the velocity change at the Moon, and this is expressed by equation (48). Observe that equation (48) involves the Kepler velocities $\mathbf{h}_v$, and an impulsive change $\Delta \mathbf{v}_s$. This constraint requires the velocity *after* the swingby to equal the vector sum of: (a) the velocity *before* the swingby, plus, (b) the velocity of the Moon, plus, (c) the impulsive change.

Objective. The objective function for this simplified model problem is to *minimize* the total Δv, i.e.

$$F = \|\Delta \mathbf{v}_1\| + \|\Delta \mathbf{v}_2\| + \|\Delta \mathbf{v}_s\| \quad (49)$$

The solution to this problem should define a geometric configuration of the Earth-Moon system that is optimal when ignoring the lunar gravitational effects.

Step 2: Three-Body Approximation

The solution to the simplified model problem computed in Step 1 is designed to construct an approximate solution to the overall problem. Unfortunately, by design, the simplified model ignores one of the primary dynamic aspects of the real problem. Specifically, the conic solution solves

$$\ddot{\mathbf{r}} = -\frac{\mu_e}{r^3}\mathbf{r} \quad (50a)$$

$$\ddot{\mathbf{r}}_L = -\frac{\mu_o}{r_L^3}\mathbf{r}_L \quad (50b)$$

but *not* the three-body dynamics

$$\ddot{\mathbf{r}} = -\frac{\mu_e}{r^3}\mathbf{r} + \mathbf{g}_L \quad (51a)$$

$$\ddot{\mathbf{r}}_L = -\frac{\mu_o}{r_L^3}\mathbf{r}_L \quad (51b)$$

Let us denote the conic solution from equation (50a) by $\tilde{\mathbf{r}}(t)$. If one assumes that the conic trajectory is approximately correct then it is reasonable to find a "nearby" trajectory that does satisfy the correct dynamics given by equation (51a). This goal can be achieved by "fitting" the three-body trajectory to the conic i.e. by minimizing

$$F = \sum_{k=1}^{N} \|\mathbf{r}_k - \tilde{\mathbf{r}}_k\|^2 \quad (52)$$

subject to

$$\ddot{\mathbf{r}} = -\frac{\mu_e}{r^3}\mathbf{r} + \mathbf{g}_L \tag{53}$$

$$\ddot{\mathbf{r}}_L = -\frac{\mu_o}{r_L^3}\mathbf{r}_L \tag{54}$$

$$r_{Lmin} \leq \|\mathbf{r} - \mathbf{r}_L\| \tag{55}$$

where $\tilde{\mathbf{r}}_k = \tilde{\mathbf{r}}(t_k)$ is the spacecraft position on the conic trajectory and $\mathbf{r}_k = \mathbf{r}(t_k)$ is the state evaluated at the same time points t_k on the three-body trajectory. Observe that in this *inverse problem* an algebraic inequality constraint equation (55) is included to insure the trajectory is sufficiently above the lunar surface where r_{Lmin} is a lower bound on the distance from the spacecraft to the Moon. The conic solution serves as a good initial guess for this inverse problem, so it is reasonable to set $\mathbf{r}_k^{(0)} = \tilde{\mathbf{r}}_k$, provided the N data points *exclude* the single time point at the Moon. Clearly the lunar acceleration $\mathbf{g}_L$ cannot be evaluated when $\|\mathbf{d}\| = \|\mathbf{r} - \mathbf{r}_L\| = \mathbf{0}$ in equation (6).

Step 3: Fixed Swingby Time

The solution obtained from Step 2 provides an excellent initial guess for Step 3. In particular, it supplies a time history for $\mathbf{r}(t)$, $\mathbf{v}(t)$ that satisfies the full three-body dynamic equations (2–5). Furthermore, by construction the boundary conditions at the park and mission orbits will be approximately satisfied. Thus one can pose a problem with two distinct phases described by either the three-body dynamic equations (2–5) or the differential-algebraic system equations (30–32). Figure 8 illustrates the dynamic simulation.

Phase 1: Outbound Transfer. The outbound transfer begins at the free initial time t_I which must satisfy the following park orbit conditions analogous to those used for Step 1:

$$\mathbf{r}_p = \mathbf{r}_o \qquad \text{Position Continuity} \tag{56}$$

$$\mathbf{v}_p = \mathbf{v}_o - \Delta\mathbf{v}_1 \qquad \text{Impulsive Velocity Change} \tag{57}$$

$$\boldsymbol{\phi}_p(\mathbf{r}_p, \mathbf{v}_p) = \mathbf{0} \qquad \text{Park Orbit Constraints} \tag{58}$$

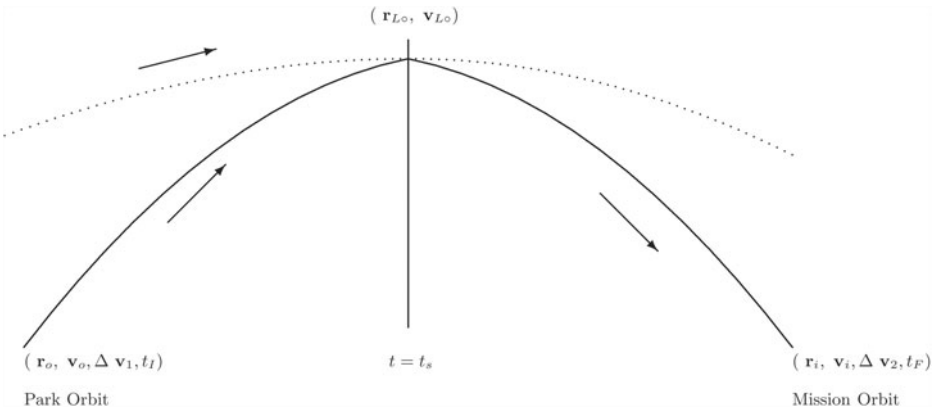


FIG. 8. Fixed Swingby Time Solution.

The phase terminates at the fixed time t_s which must satisfy the lunar conditions

$$\|\mathbf{r} - \mathbf{r}_L\| \geq r_{L\,min} \qquad \text{Closest Approach} \tag{59}$$

$$(\mathbf{v} - \mathbf{v}_L)^{\mathsf{T}}(\mathbf{r} - \mathbf{r}_L) = 0 \qquad \text{Lunar Flight Path Angle} \tag{60}$$

Observe that by forcing the lunar flight path angle to be zero equation (60) insures that the closest approach to the Moon will occur at t_s.

Phase 2: Inbound Transfer. The inbound transfer begins at the fixed time t_s and must satisfy the conditions

$$(\mathbf{r}, \mathbf{v}, \mathbf{r}_L, \mathbf{v}_L)^{(-)} = (\mathbf{r}, \mathbf{v}, \mathbf{r}_L, \mathbf{v}_L)^{(+)} \qquad \text{State Continuity} \tag{61}$$

$$(\mathbf{r}_L, \mathbf{v}_L) = (\bar{\mathbf{r}}_L, \bar{\mathbf{v}}_L) \qquad \text{Lunar State} \tag{62}$$

Equation (61) ensures continuity in all of the states across the phase boundary, and since the time t_s is fixed equation (62) enforces consistency with the lunar reference epoch $(\bar{\mathbf{r}}_L, \bar{\mathbf{v}}_L)$.

Phase 2 terminates at the free time t_F and at that point must satisfy the mission orbit conditions

$$\mathbf{r}_m = \mathbf{r}_i \qquad \text{Position Continuity} \tag{63}$$

$$\mathbf{v}_m = \mathbf{v}_i + \Delta\mathbf{v}_2 \qquad \text{Impulsive Velocity Change} \tag{64}$$

$$\boldsymbol{\phi}_m(\mathbf{r}_m, \mathbf{v}_m) = \mathbf{0} \qquad \text{Mission Orbit Constraints} \tag{65}$$

$$t_{max} \geq t_F - t_I \qquad \text{Mission Duration} \tag{66}$$

The objective for this dynamic optimization problem is to minimize

$$F = \|\Delta\mathbf{v}_1\| + \|\Delta\mathbf{v}_2\| \tag{67}$$

Step 4: Optimal Three-Body Solution

The completion of Step 3 will provide an excellent initial guess for the full optimal three-body lunar swingby. Indeed, only two modifications to the formulation in Step 3 are required. First, of course, the time of closest approach to the Moon, t_s, must be free. Second, one must ensure that the lunar state at the (free) initial time is consistent with the reference epoch for the Moon. Thus at the beginning of Phase 1, the following additional boundary conditions must be satisfied:

$$\mathbf{r}_L = \mathbf{h}_r(\mathbf{r}_o, \mathbf{v}_o, \Delta E_o) \tag{68}$$

$$\mathbf{v}_L = \mathbf{h}_v(\mathbf{r}_o, \mathbf{v}_o, \Delta E_o) \tag{69}$$

Typically any convenient reference epoch for the Moon can be chosen.

Solving the Subproblems

The preceding sections posed a set of subproblems that can be solved to obtain an optimal lunar swingby trajectory. But how does one efficiently solve the subproblems? In particular it is necessary to solve:

- *an optimal control problem* in Steps 3 and 4.
- *a parameter estimation (inverse) problem* in Step 2.
- *a nonlinear programming problem* in Step 1.

There are many good algorithms and corresponding software implementations available to solve the nonlinear programming (NLP) problem:

Find *variables* $\mathbf{x}^{\mathsf{T}} = (x_1, \ldots, x_n)$
to minimize the *objective*

$$F(\mathbf{x})$$

subject to *constraints*

$$\mathbf{c}_L \leq \mathbf{c}(\mathbf{x}) \leq \mathbf{c}_U.$$

However, the problems we want to solve are of the form

Find *control functions* $\mathbf{u}(t)$ and/or parameters $\mathbf{p}$ to minimize

$$J = \int_{t_1}^{t_F} w[\mathbf{y}(t), \mathbf{u}(t), \mathbf{p}, t]\,\mathrm{d}t$$

or

$$J = \sum_k \|\mathbf{y}(t_k) - \tilde{\mathbf{y}}(t_k)\|^2$$

subject to constraints over the domain $t_I \leq t \leq t_F$

$$\dot{\mathbf{y}} = \mathbf{f}[\mathbf{y}(t), \mathbf{u}(t), \mathbf{p}, t]$$
$$\mathbf{0} \leq \mathbf{g}[\mathbf{y}(t), \mathbf{u}(t), \mathbf{p}, t]$$

with associated boundary conditions.

So why is this problematic? The NLP methods work with a *finite* set of variables $\mathbf{x}$ and functions $f(\mathbf{x})$, $\mathbf{c}(\mathbf{x})$. However, both the optimal control and estimation problems are *infinite* dimensional, i.e. they involve the functions $\mathbf{u}(t)$ and $\mathbf{y}(t)$. How do we formulate the problem?

Historically *shooting methods* were employed to in effect "eliminate" the infinite dimensional problem by solving

$$\dot{\mathbf{y}} = \mathbf{f}[\mathbf{y}(t), \mathbf{u}(t), t] \qquad \text{and/or} \qquad \begin{aligned} \dot{\mathbf{y}} &= \mathbf{f}[\mathbf{y}(t), \mathbf{u}(t), t] \\ \mathbf{0} &= \mathbf{g}[\mathbf{y}(t), \mathbf{u}(t), t] \end{aligned}$$

Using this approach the NLP involves the *finite* set of boundary values treated as optimization variables. Unfortunately, the resulting boundary value problem is *very* nonlinear. For the swingby trajectory the Δv variables appear at the problem boundaries, whereas the lunar gravity appears as a perturbation in the "middle" of the problem. As a consequence the resulting ODE or DAE can be very unstable. Furthermore because it is necessary to incorporate error control mechanisms in order to numerically propagate the equations of motion, and this must be done during all

optimization iterations the overall process is very inefficient. Treatment of path inequality constraints is at best cumbersome and often simply impractical using a shooting method. In effect a shooting method which forces the dynamic constraints (ODE's) to be satisfied at every iteration is analogous to a generalized reduced gradient (GRG) algorithm for solving an NLP. It is generally recognized that a "constraint following" method is very inefficient. In short, a shooting method is not an attractive alternative for this problem.

Discretization Methods utilize an entirely different approach. The dynamic variables $\mathbf{y}(t)$, $\mathbf{u}(t)$ are discretized leading to a finite, albeit large, set of variables. The differential equations or dynamic constraints are then replaced by nonlinear algebraic constraints. Figure 9 illustrates the approach using a very simple trapezoidal rule to approximate the ODE. In general there are three basic operations when using a so-called *direct transcription method*. Specifically the approach is as follows:

An Optimal Control Algorithm

Direct Transcription: Transcribe the optimal control problem into a nonlinear programming (NLP) problem by discretization.

Sparse Nonlinear Program: Solve the sparse (SQP or Barrier) NLP.

Mesh Refinement: Assess the accuracy of the approximation (i.e. the finite dimensional problem), and if necessary refine the discretization, and then repeat the optimization steps.

In fact, the approach can be considered an SNLP (Sequential Nonlinear Programming Algorithm). All of the relevant techniques for solving optimal control problems using direct transcription have been implemented in the $\mathbb{SOCS}$ software, and are discussed in [3], [5], and [4]. The methodology is extended to inverse problems in [7], by utilizing NLP methods appropriate for large scale least squares applications.

Barrier or SQP Algorithm?

There are a number of efficiency issues that must be addressed within the context of an SNLP algorithm. The second step of the algorithm requires the solution of an NLP, however, among the many possible nonlinear programming algorithms it is desirable to choose the most efficient. Computational experience provides some insight to guide the choice. Specifically, let us compare the performance of two different

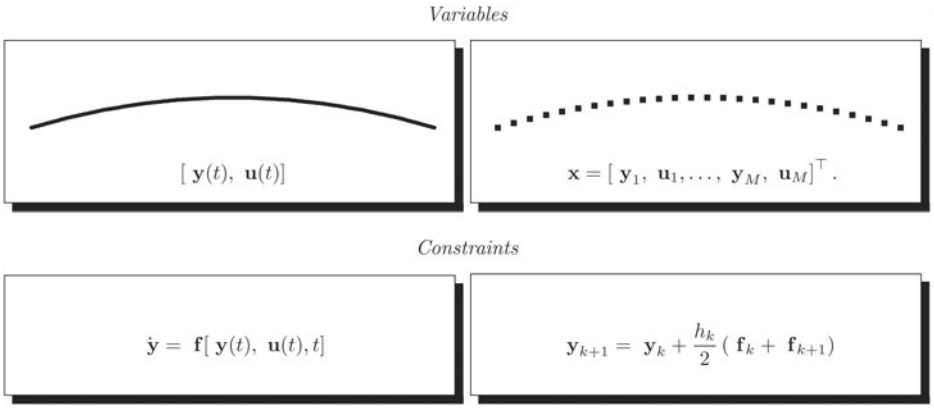


FIG. 9. Discretization Methods.

NLP algorithms when solving the small, dense NLP subproblems required by Step 1 from the previous section. Table 1 summarizes the performance of two different NLP algorithms when used to solve three different NLP problems. In particular results are presented for the sequential quadratic programming (SQP) algorithm described in [3] and [5]. Two different techniques were used to compute the Hessian matrix, namely a finite difference approximation denoted as "Newton," and a quasi-Newton "BFGS" (Broyden-Fletcher-Goldfarb-Shanno) recursive update. The second algorithm is the primal-dual interior point or barrier algorithm described in [4]. This algorithm was also tested using Newton and BFGS Hessian approximations.

For each algorithm the table presents the number of gradient and Hessian evaluations needed to reach a solution. Although the number of problems in this test set is quite small, the basic findings are consistent with much more extensive testing as described in [6]. Generally the SQP algorithm was both more efficient and more robust than the barrier method. Although it is difficult to prove, one speculates that an SQP method is simply a better choice for solving very nonlinear optimization problems. Conversely, our experience suggests a barrier algorithm may be preferable for problems with linear constraints, especially when there are many inequalities. The testing also suggests that a quasi-Newton Hessian approximation requires significantly more iterations to converge. While this is not surprising, it is extremely important when considering the very large sparse NLP problems that arise when using discretization methods for optimal control.

A quasi-Newton Hessian approximation suffers from another deficiency that is not demonstrated by this comparison. The results given utilize a quasi-Newton approximation to the *full* Hessian. However, exploiting matrix sparsity when using a quasi-Newton update has to date been computationally unsuccessful. One common alternative is to apply a quasi-Newton update to the *projected or reduced* Hessian, which is dense. For example this technique is used by Gill, Murray, and Saunders [11] in the software SNOPT. Unfortunately, as they state in [12] the method is ". . . best suited for problems with a moderate number of degrees of freedom (say, up to 2000)." Byrd, Hribar, and Nocedal [8] use a *limited memory update* in the software KNITRO in an attempt to deal with the storage requirements of the dense projected Hessian matrix. Unfortunately like all quasi-Newton methods, limited memory updates do not demonstrate quadratic convergence, and consequently become prohibitively expensive for problems with many degrees of freedom. Generally, algorithms that fully exploit Hessian sparsity have demonstrated superior computational performance for large scale optimization. Full Hessian sparsity is exploited by the SQP and barrier methods in SOCS, as well as the LOQO algorithm of Vanderbei and Shanno [15].

TABLE 1. NLP Algorithm Performance Comparison

Step 1-Small, Dense NLP Subproblems			
Mission	Equatorial	Polar	Molniya
SQP-Newton	(10,4)	(16,10)	(135,44)
SQP-BFGS	(24,19)	(36,31)	(186,96)
Barrier-Newton	(24,22)	(57,55)	(70,68)†
Barrier-BFGS	(242,241)†	(58,56)	(286,284)

Key: (Gradient Eval., Hessian Eval.) †No Solution

A second more serious performance issue occurs when comparing NLP algorithms for use within the context of an SNLP. Is an SQP better than a barrier method for Sequential Nonlinear Programming? Typically the NLP problem size grows as the discretization mesh is refined. In this context it is required that one efficiently solve a *sequence* of NLP's. The obvious way to achieve this goal is to use coarse grid information to "hot start" the fine grid NLP. In fact one can use high order polynomial interpolation of the coarse grid solution to construct a very good initial guess for the NLP problem that must be solved on a fine grid. Thus, as the mesh is refined the initial guess for the NLP subproblem becomes better and better. An SQP algorithm can exploit this. In contrast for an interior point algorithm, the iterates must be *strictly feasible.* Constructing a feasible initial iterate by perturbing the user supplied initial guess is a straightforward process performed by computational software. However in so doing, the very first iterate for each barrier NLP is inconsistent with the coarse grid interpolation. In short, a *barrier algorithm cannot exploit a good guess!* This fundamental shortcoming of an interior point method is discussed by Forsgren [10].

To illustrate this point consider the solution of the optimal control subproblem for the polar mission summarized in Tables 2 and 3. Both cases were initiated using the same information. Specifically the initial trajectory was constructed as the solution from a Step 2 inverse problem. Using this three-body solution trajectory a variable step-size numerical integration algorithm was used to construct the initial grid points. Referring to Table 2, the first grid had 594 points, leading to an NLP problem with 7136 optimization variables and 7715 nonlinear constraints. The solution to this coarse grid problem requires 18 gradient evaluations (NGC), 10 Hessian evaluations (NHC), and 3794 function evaluations (NFE) including those needed for finite difference derivatives. The resulting solution had a relative discretization error of $\epsilon = 1 \times 10^{-4}$ and was computed in 30.1 seconds. The grid was refined two times, with the final grid containing 1113 points. Table 3 presents exactly the same history, when a barrier algorithm is used to solve the NLP subproblems. For the SQP algorithm as the mesh is refined, the number of Hessian evaluations and iterations decreases—only one Hessian is needed on the second on third grids. Each coarse grid solution provides a very good guess, and the Newton method is within its region of quadratic convergence. For the barrier method this is not true! The second mesh required an additional 49 Hessian evaluations because the initial guess was perturbed. The overall penalty in computation time is catastrophic in this example. In addition, because the initial guesses were perturbed the barrier algorithm converged to a different local solution than the SQP. All of the computational results were obtained using a Dell M60 laptop computer, with a Linux operating system.

TABLE 2. Mesh Refinement with an SQP Algorithm

				SQP				
k	M	n	m	NGC	NHC	NFE	ϵ	Time (sec)
1	594	7136	7715	18	10	3794	1×10^{-4}	30.1
2	881	10580	11446	4	1	454	4×10^{-7}	7.8
3	1113	13364	14462	4	1	454	1×10^{-8}	10.6
Total				26	12	4702		48.6

TABLE 3. Mesh Refinement with a Barrier Algorithm

				Barrier†				
k	M	n	m	NGC	NHC	NFE	ϵ	Time (sec)
1	594	7720	7715	328	319	112475	1×10^{-5}	749.3
2	881	12985	12980	57	49	17637	2×10^{-7}	233.7
3	1113	14090	14085	6	2	881	1×10^{-8}	18.5
Total				391	370	130993		1001.6

†Different Local Solution than SQP.

Is Mesh Refinement Needed?

In light of the apparent conflict between mesh refinement and a barrier algorithm it is important to review why it is necessary. Let us consider the solution of the Step 2 inverse problem solution for the polar mission. To review, the conic trajectory available from the solution of Step 1 can be used to construct an initial guess. In particular by sampling the conic at 600 equal ΔE increments, one obtains a nonlinear least squares problem with 1800 residuals. To avoid a singularity it is necessary to omit the point at the Moon. The shaded region in Fig. 10 illustrates the discontinuous behavior in one component of the velocity from the conic trajectory, and the solid line shows the smoothed approximation that results after "fitting" a three-body

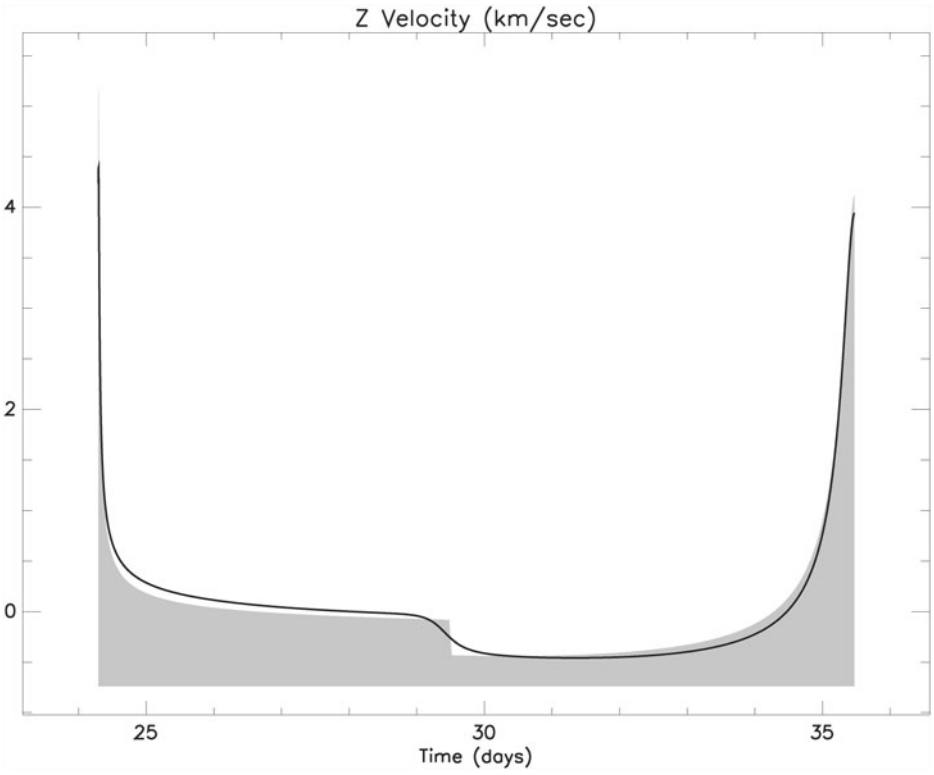


FIG. 10. Velocity Discontinuity.

TABLE 4. Mesh Refinement

k	M	n	m	NGC	NHC	NFE	ϵ	Time (sec)
1	599	7188	7775	12	2	144	6×10^{-1}	3.5
2	606	7272	7866	21	17	525	9×10^{-5}	1.1
3	606	7272	7866	4	2	724	1×10^{-6}	7.6
4	742	8904	9634	4	1	448	1×10^{-8}	5.6
Total				41	22	1841		27.7

k	Refinement No	M	Grid Pts	n	NLP vars
m	NLP cons.	NGC	Grad Eval	NHC	Hess Eval
NFE	Func Eval	ϵ	Disc. Error	Time	CPU

solution to the conic. Table 4 summarizes the behavior of the SOCS mesh refinement procedure for this example and Fig. 11 illustrates what the procedure does to both the discretization error and the mesh distribution. Initially the discretization error is very large in the vicinity of the Moon. Clearly this error can be attributed to the approximate nature of the conic trajectory—i.e. the position goes through the center of the Moon and the velocity change is impulsive. During the first few refinement iterations grid points are added in the vicinity of the discontinuity and this leads to a significant reduction in the discretization error as measured by the value of ϵ in Table 4. In fact after the first refinement iteration only six grid points were added, all in the neighborhood of the discontinuity, and this reduced the discretization error by nearly four orders of magnitude. It is also worth noting that solving the NLP subproblem after adding these grid points was significantly more expensive (taking 17 Hessian evaluations), because the entire solution had to be adjusted to account for this effect. Clearly, mesh refinement is needed in order to address the singularities in the vicinity of the Moon.

DAE or ODE Formulation?

The three-body dynamics of the system can be described by the ordinary differential equations summarized in the third section. However, the differential-algebraic system presented in the fifth section can also be used when solving the subproblems

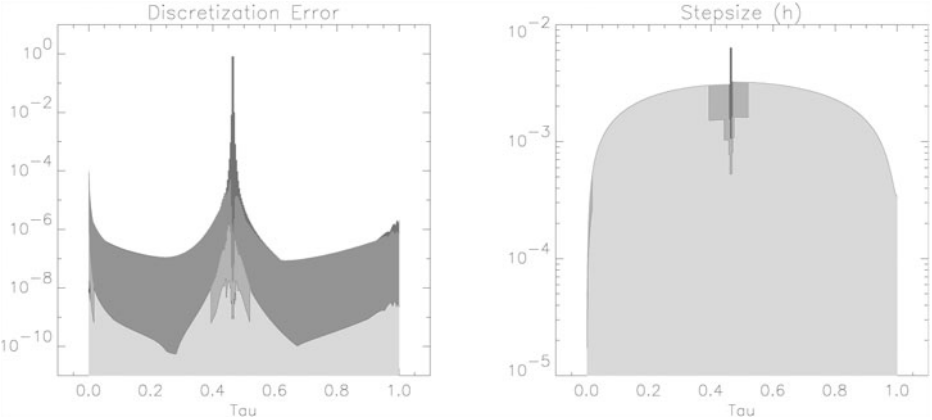


FIG. 11. Mesh Refinement.

TABLE 5. Steps 3 & 4, Optimal Solution for Molniya Mission

ODE Formulation				
M	*n*	NGC	NHC	Time (sec)
630	7568	26	7	32.02
807	9692	7	2	12.30
940	11288	4	1	8.66
1190	14288	4	1	11.22
1190	14291	10	4	45.49

Total Time = 109.69 sec

TABLE 6. Steps 3 & 4, Optimal Solution for Molniya Mission

DAE Formulation				
M	*n*	NGC	NHC	Time (sec)
1097	8782	30	14	45.27
1530	12246	4	1	7.98
2056	16454	4	1	11.82
2056	16456	7	2	31.02

Total Time = 96.09 sec

in Steps 2, 3, and 4. Is one formulation preferable to the other in terms of either computational speed and/or solution accuracy? To address this question, Tables 5 and 6 compare the formulations. Specifically, the optimal control problems in Step 3 and 4, for the Molniya mission were solved using the ODE and DAE formulations. Both were initialized with the same Step 2 solution trajectory.

The comparison reveals a number of issues. First, the DAE formulation required a final grid with 2056 points, whereas the ODE formulation achieved the same accuracy with 1190 points. There are 12 ODE's and the DAE system has only seven equations. Since the number of grid points is related to the nonlinearity of the equations as well as the order of interpolation, this suggests the DAE system may be more nonlinear. However, the total solution time is dictated by the total number of NLP variables in the discretized subproblem. The ODE formulation requires 14,291 variables for the final iteration. In contrast, the DAE formulation needs 16,456 variables. Thus the size of the NLP problems is nearly the same, even though one formulation involves nearly twice as many dynamic equations. Overall, there was no clearcut difference between the ODE and DAE formulations in either speed or accuracy for the applications considered here.

Summary and Conclusions

This paper demonstrates that optimal lunar swingby trajectories can yield significant performance benefits for Earth orbital missions that require large plane change. A four-step technique for computing the orbit transfers was presented and demonstrated on a number of example missions. The approach requires first computing an

approximate solution using simple two-body dynamics. The conic solution is then used to construct a three-body trajectory by incorporating a large scale parameter estimation or inverse problem technique. The three-body trajectory serves as an excellent initial guess for the final two trajectory optimization steps. Although the stepwise procedure is illustrated for lunar swingby trajectories it has more general applicability for any interplanetary mission analysis that currently employs "patched conic" techniques.

The lunar swingby trajectory optimization problem is a very *nonlinear boundary value problem.* As such, it serves as a challenging application to use when testing various numerical solution techniques. Simple shooting methods are ineffective because the lunar gravitational effects appear in the "middle" of the problem and tend to accentuate errors propagated from the trajectory boundary. In contrast, large scale collocation methods demonstrate the necessary robustness to deal with these instabilities. Within the context of direct transcription methods, it is demonstrated that a sequential quadratic programming (SQP) method is more robust and efficient than a primal-dual barrier algorithm. In particular, a barrier algorithm cannot exploit a good guess, which has a significant impact on both efficiency and robustness. Finally, the use of an efficient mesh refinement procedure is critical for computing a stable solution to the problem.

References

[1] BATTIN, R. H. *An Introduction to the Mathematics and Methods of Astrodynamics,* AIAA Education Series, American Institute of Aeronautics and Astronautics, Inc., 1633 Broadway, New York, NY 10019, 1987.

[2] BETTS, J. T. "Optimal Three-Burn Orbit Transfer," *AIAA Journal,* Vol. 15, No. 6, 1977, pp. 861–864.

[3] *Practical Methods for Optimal Control using Nonlinear Programming,* Society for Industrial and Applied Mathematics, Philadelphia, PA., 2001.

[4] BETTS, J. T., ELDERSVELD, S. K., FRANK, P. D., and LEWIS, J. G. "An Interior-Point Nonlinear Programming Algorithm for Very Large Scale Optimization," in *Large-Scale PDE-Constrained Optimization,* L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, eds., Springer-Verlag, Berlin, 2003, pp. 184–198.

[5] BETTS, J. T. and FRANK, P. D. "A Sparse Nonlinear Optimization Algorithm," *Journal of Optimization Theory and Applications,* Vol. 82, No. 3, 1994, pp. 519–541.

[6] BETTS, J. T. and GABLONSKY, J. M. "A Comparison of Interior Point and SQP Methods on Optimal Control Problems" Technical Document Series MCT-TECH-02-004, Mathematics and Engineering Analysis, The Boeing Company, PO Box 3707, Seattle, WA 98124-2207, Mar. 2002.

[7] BETTS, J. T. and HUFFMAN, W. P. "Large Scale Parameter Estimation Using Sparse Nonlinear Programming Methods," *SIAM Journal on Optimization,* Vol. 14, No. 1, 2003, pp. 198–206.

[8] BYRD, R. H., HRIBAR, M. E., and NOCEDAL, J. "An Interior Point Algorithm for Large Scale Nonlinear Programming," *SIAM Journal on Optimization,* Vol. 9, 2000, pp. 877–900.

[9] ESCOBAL, P. R. *Methods of Orbit Determination,* Second Edition, Robert E. Krieger Publishing Company, Malabar, Florida, 1985.

[10] FORSGREN, A. "On Warm Starts for Interior Methods," in *System Modeling and Optimization,* IFIP International Federation for Information Processing, F. Ceragioli, A. Dontchev, H. Furuta, K. Marti and L. Pandolfi, eds., vol. 199, Springer, Boston, 2006, pp. 51–66.

[11] GILL, P. E., MURRAY, W., and SAUNDERS, M. A. "SNOPT: An SQP Algorithm for Large-scale Constrained Optimization," Technical Report SOL 97-3, Department of Operations Research, Stanford University, 1997.

[12] GILL, P. E., MURRAY, W., and SAUNDERS, M. A. "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review,* Vol. 47, 2005, pp. 99–131.

[13] LEE, T., DUNHAM, D., HSU, S., and ROBERTS, C. "Large Inclination Change Using Lunar-Swingby Technique," presented as paper AIAA 1988-9290 at the AIAA/AAS Astrodynamics Conference, Minneapolis, MN, Aug. 1988.

[14] STANDISH, E. M. "JPL Planetary and Lunar Ephemerides, DE405/LE405," Interoffice Memorandum TOM 312.F-98-048, Jet Propulsion Laboratory, Pasadena, CA, Aug. 1998.
[15] VANDERBEI, R. J., and SHANNO, D. F. "An Interior-Point Method for Nonconvex Nonlinear Programming," *Computational Optimization and Applications,* Vol. 13, 1999, pp. 231–252.
[16] WILSON, R. S., and HOWELL, K. C. "Trajectory Design in the Sun-Earth-Moon System Using Lunar Gravity Assists," *Journal of Spacecraft and Rockets,* Vol. 35, 1998, pp. 191–198.

Appendix A: Reference Epoch

The reference epoch for all numerical results corresponds to a Julian date of 2453561.5, (July 10, 2005). All results utilize an ICRF (International Celestial Reference Frame). The lunar ephemeris was constructed using a least squares fit of two-body dynamics to the JPL DE405 lunar ephemeris [14] over a 60 day period beginning at the reference epoch. The resulting lunar state vector, and corresponding equatorial radii and gravitational constants are given in Table 7.

TABLE 7. Mission Parameters

μ_e	398600.436380820 km^3/sec^2
R_e	6378.14000000000 km
μ_L	4902.79881586123 km^3/sec^2
R_L	1737.40000000000 km
r_{ox}	$-0.3398817123704749 \times 10^6$ km
r_{oy}	$0.1956358580374241 \times 10^6$ km
r_{oz}	$0.1139974125070158 \times 10^6$ km
v_{ox}	$-0.5195857465292167 \times 10^0$ km/sec
v_{oy}	$-0.7186784200912856 \times 10^0$ km/sec
v_{oz}	$-0.3742849669631482 \times 10^0$ km/sec