Chapter 6

Optimal Control Examples

6.1 Space Shuttle Reentry Trajectory

Construction of the reentry trajectory for the space shuttle is a classic example of an optimal control problem. The problem is of considerable practical interest and is nearly intractable using a simple shooting method because of its nonlinear behavior. Early results were presented by Bulirsch [55] on one version of the problem, as well by Dickmanns [73]. Ascher, Mattheij, and Russell present a similar problem [2, p. 23] and Brenan, Campbell, and Petzold discuss a closely related path control problem [48, p. 157]. Let us consider a particular variant of the problem originally described in [175].

The motion of the vehicle is defined by the following set of DAEs:

$$\dot{h} = v \sin\gamma, \tag{6.1}$$

$$\dot{\phi} = \frac{v}{r} \cos\gamma \sin\psi / \cos\theta, \tag{6.2}$$

$$\dot{\theta} = \frac{v}{r} \cos\gamma \cos\psi, \tag{6.3}$$

$$\dot{v} = -\frac{D}{m} - g \sin\gamma, \tag{6.4}$$

$$\dot{\gamma} = \frac{L}{mv} \cos(\beta) + \cos\gamma \left(\frac{v}{r} - \frac{g}{v} \right), \tag{6.5}$$

$$\dot{\psi} = \frac{1}{mv\cos\gamma} L \sin(\beta) + \frac{v}{r\cos\theta} \cos\gamma \sin\psi \sin\theta, \tag{6.6}$$

$$q \le q_U, \tag{6.7}$$

where the aerodynamic heating on the vehicle wing leading edge is $q = q_a q_r$ and the dynamic variables are

h	altitude (ft),	γ	flight path angle (rad),
ϕ	longitude (rad),	ψ	azimuth (rad),
θ	latitude (rad),	α	angle of attack (rad),
v	velocity (ft/sec),	β	bank angle (rad).

For the sake of reference, the aerodynamic and atmospheric forces on the vehicle are specified by the following quantities (English units):

$$
\begin{aligned}
D &= \frac{1}{2} c_D S \rho v^2, & a_0 &= -0.20704,\\
L &= \frac{1}{2} c_L S \rho v^2, & a_1 &= 0.029244,\\
g &= \mu / r^2, & \mu &= 0.14076539 \times 10^{17},\\
r &= R_e + h, & b_0 &= 0.07854,\\
\rho &= \rho_0 \exp[-h/h_r], & b_1 &= -0.61592 \times 10^{-2},\\
\rho_0 &= 0.002378, & b_2 &= 0.621408 \times 10^{-3},\\
h_r &= 23800, & q_r &= 17700 \sqrt{\rho} (0.0001 v)^{3.07},\\
c_L &= a_0 + a_1 \hat{\alpha}, & q_a &= c_0 + c_1 \hat{\alpha} + c_2 \hat{\alpha}^2 + c_3 \hat{\alpha}^3,\\
c_D &= b_0 + b_1 \hat{\alpha} + b_2 \hat{\alpha}^2, & c_0 &= 1.0672181,\\
\hat{\alpha} &= 180 \alpha / \pi, & c_1 &= -0.19213774 \times 10^{-1},\\
R_e &= 20902900, & c_2 &= 0.21286289 \times 10^{-3},\\
S &= 2690, & c_3 &= -0.10117249 \times 10^{-5}.
\end{aligned}
$$

The reentry trajectory begins at an altitude where the aerodynamic forces are quite small with a weight of $w = 203000$ (lb) and mass $m = w/g_0$ (slug), where $g_0 = 32.174$ (ft/sec^2). The initial conditions are as follows:

$$
\begin{aligned}
h &= 260000 \text{ ft}, & v &= 25600 \text{ ft/sec},\\
\phi &= 0 \text{ deg}, & \gamma &= -1 \text{ deg},\\
\theta &= 0 \text{ deg}, & \psi &= 90 \text{ deg}.
\end{aligned}
$$

The final point on the reentry trajectory occurs at the unknown (free) time t_F, at the so-called terminal area energy management (TAEM) interface, which is defined by the conditions

$$
h = 80000 \text{ ft}, \qquad v = 2500 \text{ ft/sec}, \qquad \gamma = -5 \text{ deg}.
$$

To obtain realistic solutions, we also restrict the trajectory by defining the following simple bounds:

$$
\begin{aligned}
0 &\le h, & -89 \text{ deg} &\le \theta \le 89 \text{ deg},\\
1 &\le v, & -89 \text{ deg} &\le \gamma \le 89 \text{ deg},\\
-90 \text{ deg} &\le \alpha \le 90 \text{ deg}, & -89 \text{ deg} &\le \beta \le 1 \text{ deg}.
\end{aligned}
$$

Example 6.1 MAXIMUM CROSSRANGE. The goal is to choose the control variables $\alpha(t)$ and $\beta(t)$ such that the final crossrange is maximized. There are many ways to define the crossrange, but for this case it is equivalent to maximizing the final latitude

$$
J = \theta(t_F). \tag{6.8}
$$

For comparison, the solution is computed with no limit on the aerodynamic heating, i.e.,

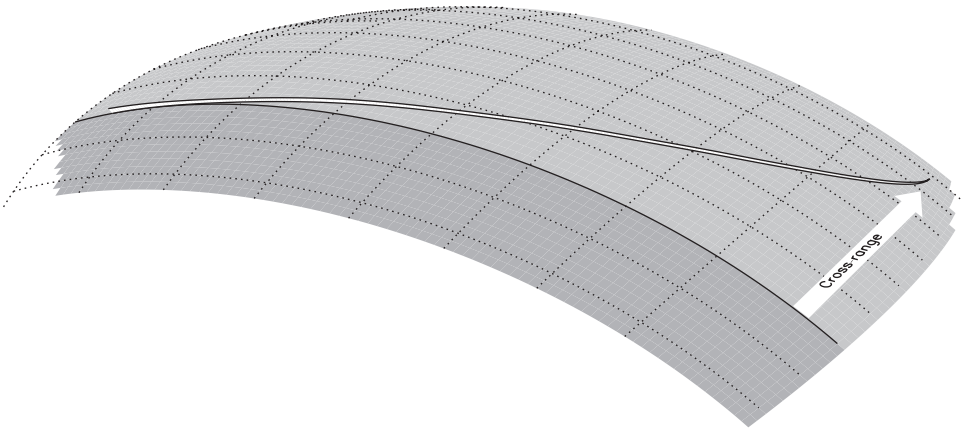


Figure 6.1. *Max crossrange shuttle reentry.*

$q_U = \infty$, and also with an upper bound on the aerodynamic heating of $q_U = 70\,\text{BTU/ft}^2\text{/sec}$. Figure 6.1 illustrates the optimal trajectory when no heating limit is imposed.

The time histories for the state are shown in Figure 6.2, with the unconstrained solution shown as a solid line and the constrained solution as a dotted line. All of the angular quantities are given in degrees, the altitude in multiples of 10^5 ft, velocity in multiples of 10^4 ft/sec, and time in seconds. In Figure 6.3, the control time histories, as well as the aerodynamic heating, are illustrated. Note that for clarity the two solutions for angle of attack are plotted with different scales—the scale corresponding to the unconstrained solution is given on the right side of the figure. The optimal values for the final time and latitude are summarized in Table 6.1. For the heat-constrained example, Figure 6.4 illustrates the behavior of the $\mathbb{SOCS}$ mesh-refinement algorithm as the discretization error is reduced below the requested tolerance of 10^{-7}. The first refinement iteration has the darkest shading and the last refinement has the lightest shading. For this case, using a linear initial guess between the boundary conditions, the solution was computed after 10 mesh-refinement iterations.

It is also interesting to ask whether mesh refinement is necessary. In order to assess the importance of mesh refinement, let us consider the following experiment. Suppose the mesh-refinement procedure is terminated after a specific number of iterations. Call the (prematurely obtained) solution $\widehat{\mathbf{u}}(t)$. This approximate "solution" can be used to propagate the trajectory, i.e., let us integrate the differential equations (6.1)–(6.6)

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \widehat{\mathbf{u}}, t) \tag{6.9}$$

from $t_I = 0$ to t_F. Let us assume the initial conditions are satisfied and assess the relative error in the terminal conditions, i.e.,

$$\epsilon = 100 \max_{k=1}^{n} \left[\frac{|\hat{y}_k(t_F) - y_k(t_F)|}{\max(|\hat{y}_k(t_F)|, |y_k(t_F)|)} \right]. \tag{6.10}$$

In order to obtain an accurate solution of the IVP, we can use any sophisticated numerical integration software to compute the value of $\widehat{\mathbf{y}}(t_F)$. We have chosen to use a variable-order, variable-stepsize Gear [90] integrator with a relative error tolerance of 10^{-14}. In essence,

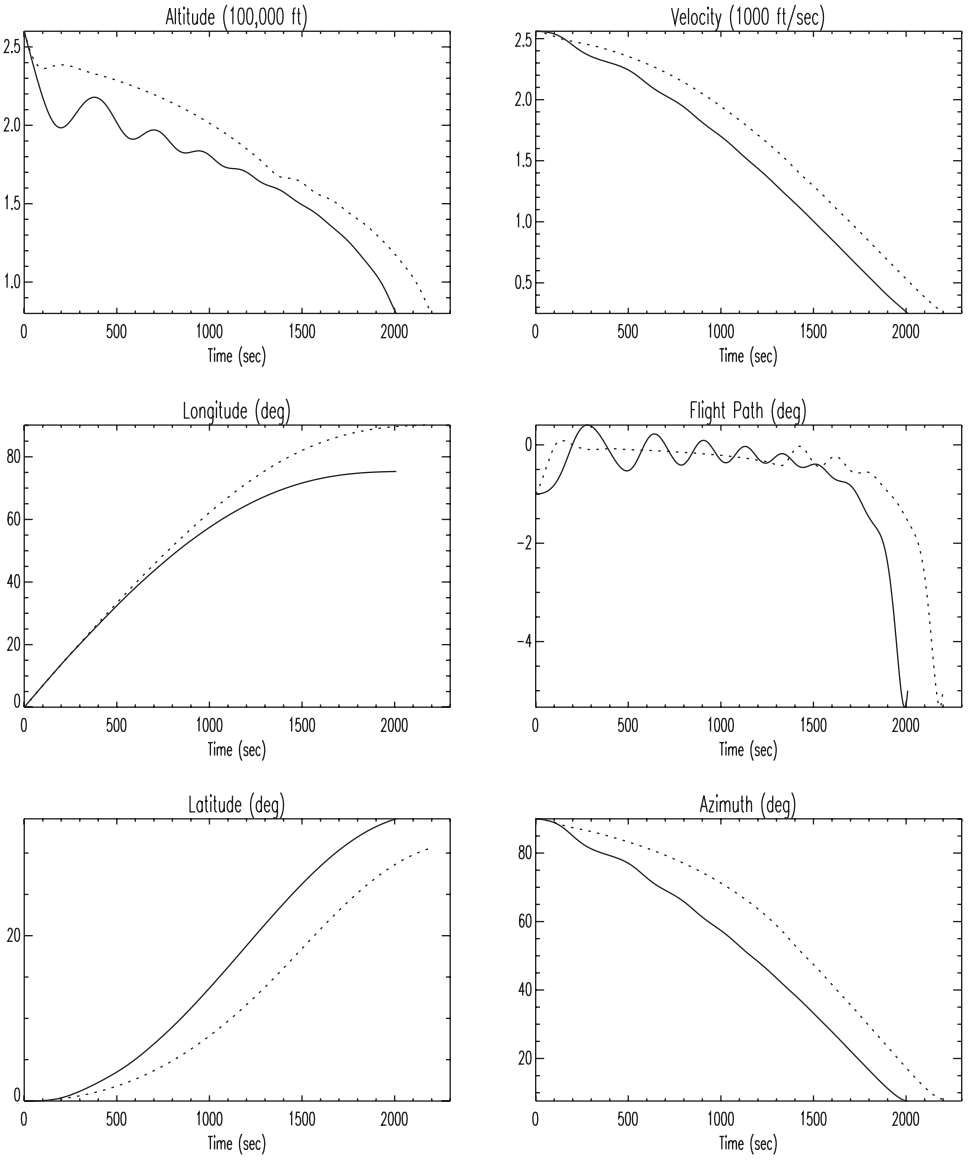


Figure 6.2. *Shuttle reentry—state variables.*

we are trying to assess how well the approximate solution $\widehat{\mathbf{u}}(t)$ to the BVP satisfies the corresponding IVP. Figure 6.5 summarizes the results of this experiment. Observe that when the approximate solution $\widehat{\mathbf{u}}(t)$ corresponds to the result after the first mesh-refinement iteration, the relative error in the integrated state is 46.5%. Thus, while the error in the objective function is only 0.4% (which might be reasonable for engineering purposes), the computed trajectory is totally unrealistic. In fact the final integrated trajectory with the approximate control $\widehat{\mathbf{u}}(t)$ has a position error of 12948.73 ft and the velocity error is 390.6111 ft/sec. Mesh refinement is crucial to obtain a realistic solution.

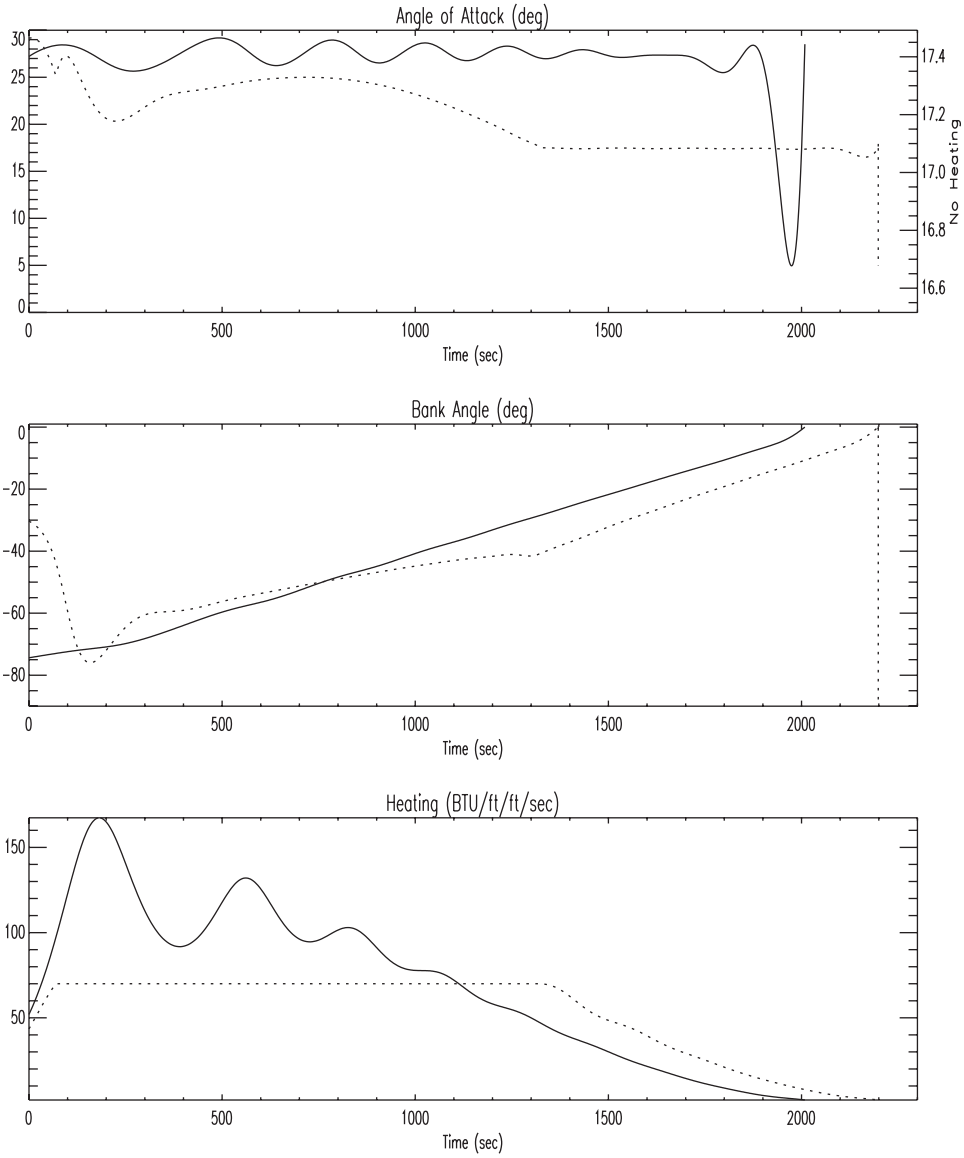


Figure 6.3. *Shuttle reentry—control variables.*

Table 6.1. *Shuttle reentry example.*

q_U	∞	70
t_F (sec)	2008.59	2198.67
$\theta(t_F)$ (deg)	34.1412	30.6255

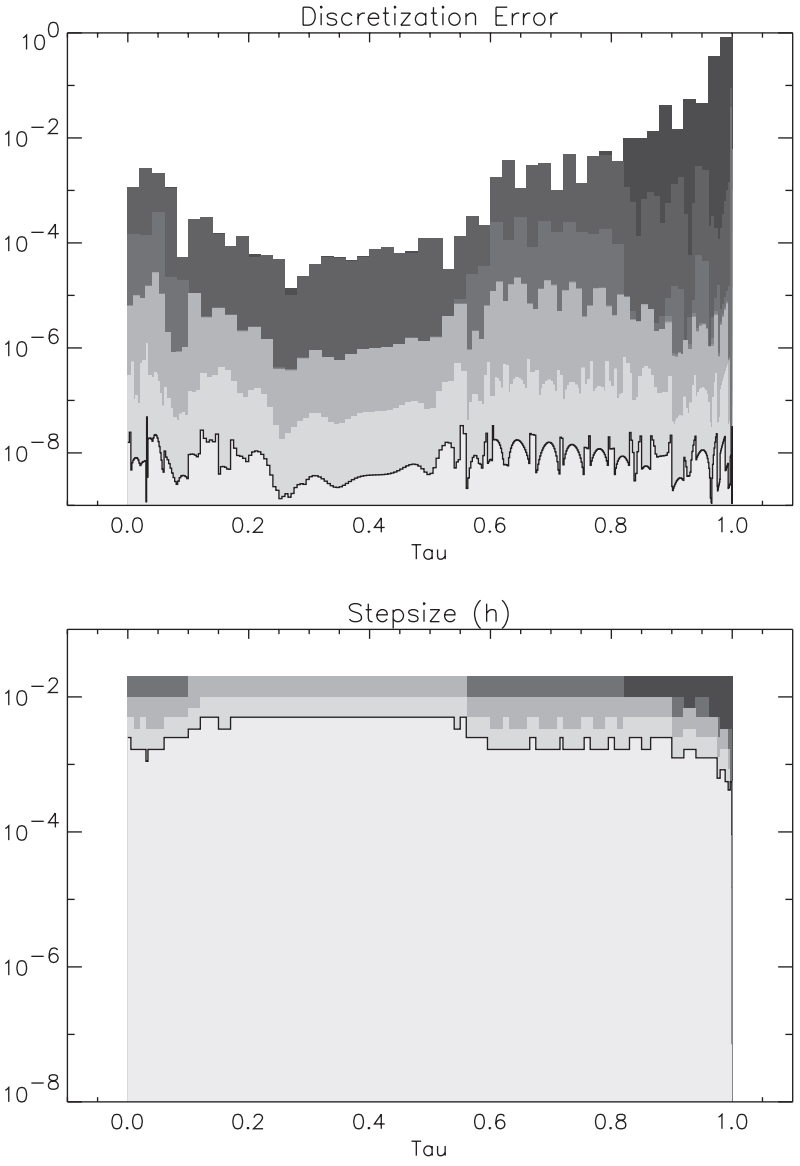


Figure 6.4. *Shuttle reentry—mesh refinement.*

Example 6.2 MAX-CROSSRANGE ALTERNATE FORMULATION. It is interesting to note that the dynamics for Example 6.1 are independent of the longitude ϕ. Observe that the longitude does not appear on the right-hand side of any of the differential equations (6.1)–(6.6). Physically this is not surprising since a spherical potential model is used to describe the earth. For convenience, in Example 6.1 the initial longitude was chosen to be $\phi = 0$. Another alternative is to simply eliminate the longitude as a state variable and the

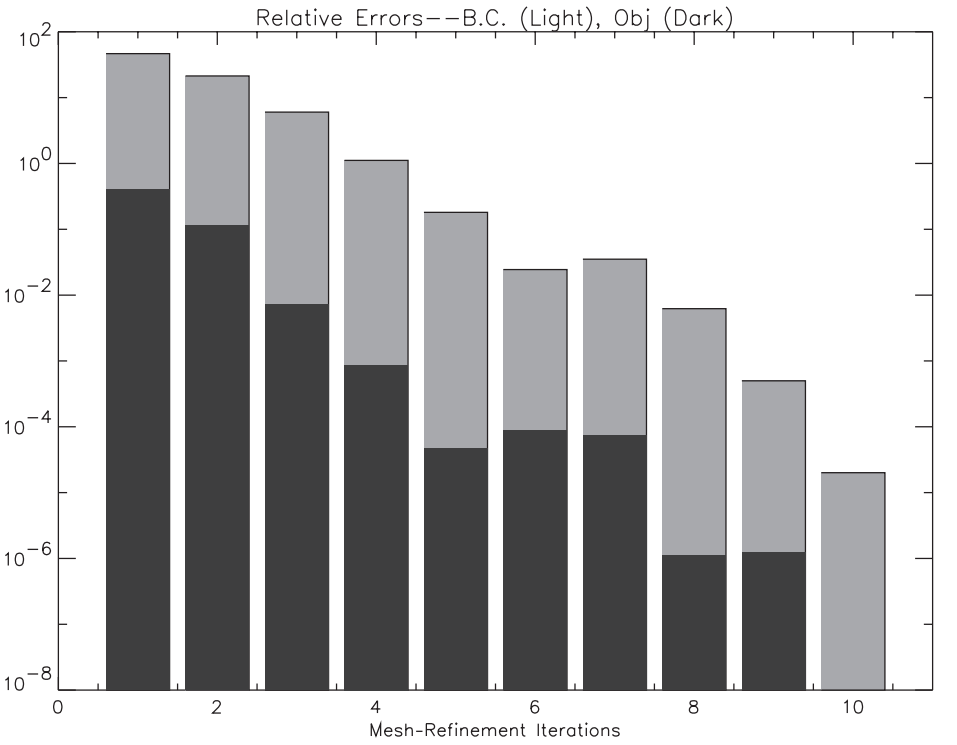


Figure 6.5. *Coarse grid solution errors.*

corresponding differential equation (6.2). Solutions to the problem using this formulation were presented in Section 4.13.

Example 6.3 MINIMAX HEATING FORMULATION. Computing a reentry profile that maximizes the crossrange (6.8) is not the only criterion that can be used. As an alternative it may be desirable to minimize the peak value of the heating q. This can be achieved by treating the upper bound q_U in (6.7) as an optimization variable. So instead of the objective (6.8) let us compute the control variables $\mathbf{u}(t)$ to minimize

$$J = q_U. \tag{6.11}$$

To preclude excessive oscillations in the solution let as also impose bounds on the rates $|\dot{\gamma}| \le \varrho$ and $|\dot{\psi}| \le \varsigma$ leading to the additional path constraints

$$-\varrho \le \quad \frac{L}{mv}\cos(\beta) + \cos\gamma\left(\frac{v}{r} - \frac{g}{v}\right) \quad \le \varrho, \tag{6.12}$$

$$-\varsigma \le \quad \frac{1}{mv\cos\gamma} L\sin(\beta) + \frac{v}{r\cos\theta}\cos\gamma\sin\psi\sin\theta \quad \le \varsigma, \tag{6.13}$$

where $\varrho = \varsigma = .2$ deg/sec. In contrast to Example 6.1, which maximized the final latitude, here let us consider three cases, with the fixed values $\theta(t_F) = 15, 20, 25$ deg. Table 6.2 summarizes the optimal solutions. Figure 6.6 illustrates the state variable history for each

Table 6.2. *Minimax heating reentry example.*

$\theta(t_F)$ (deg)	t_F (sec)	q_U^*
25	1994.44	49.8777
20	1714.81	38.0550
15	1390.80	27.9982

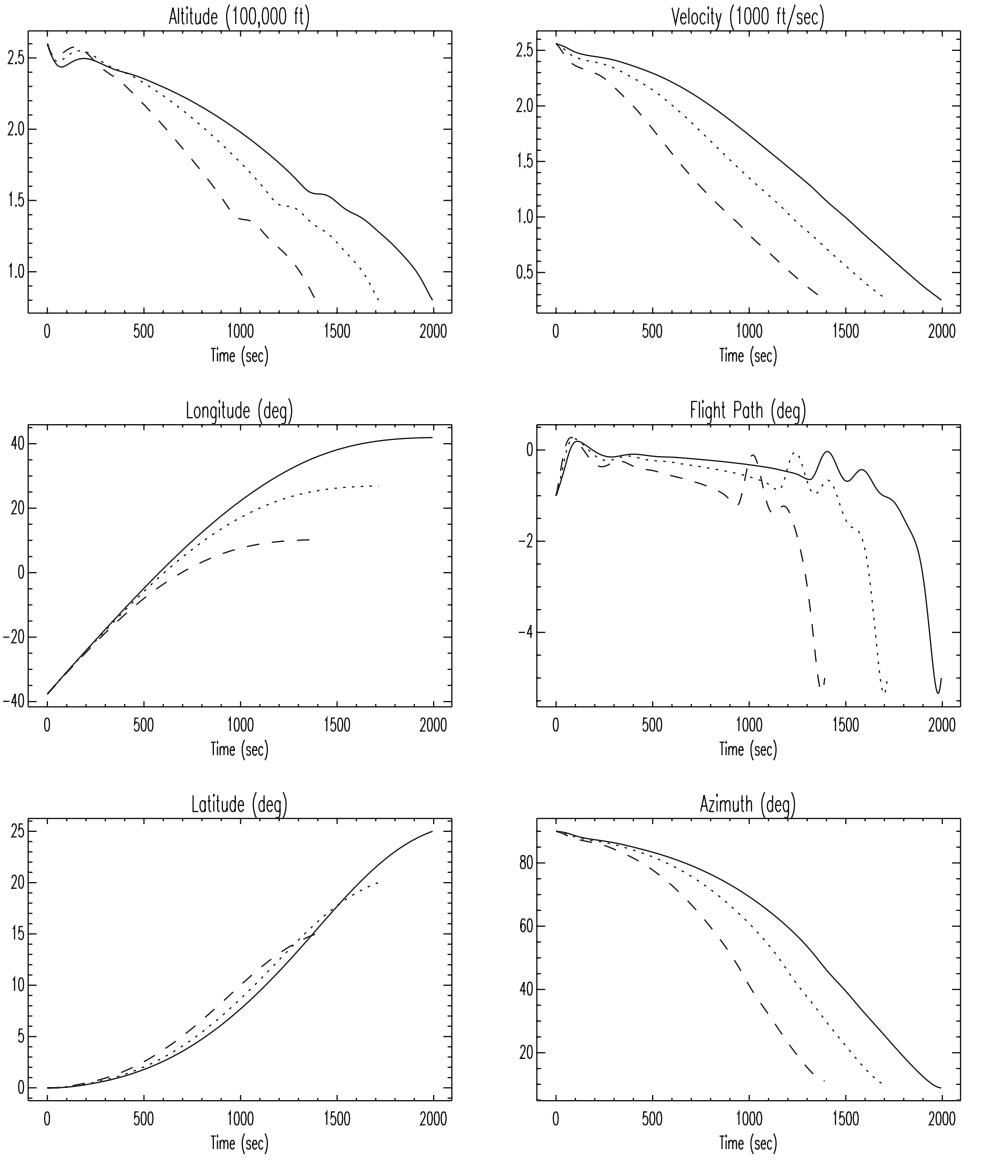


Figure 6.6. *Minimax heating shuttle reentry—state variables.*

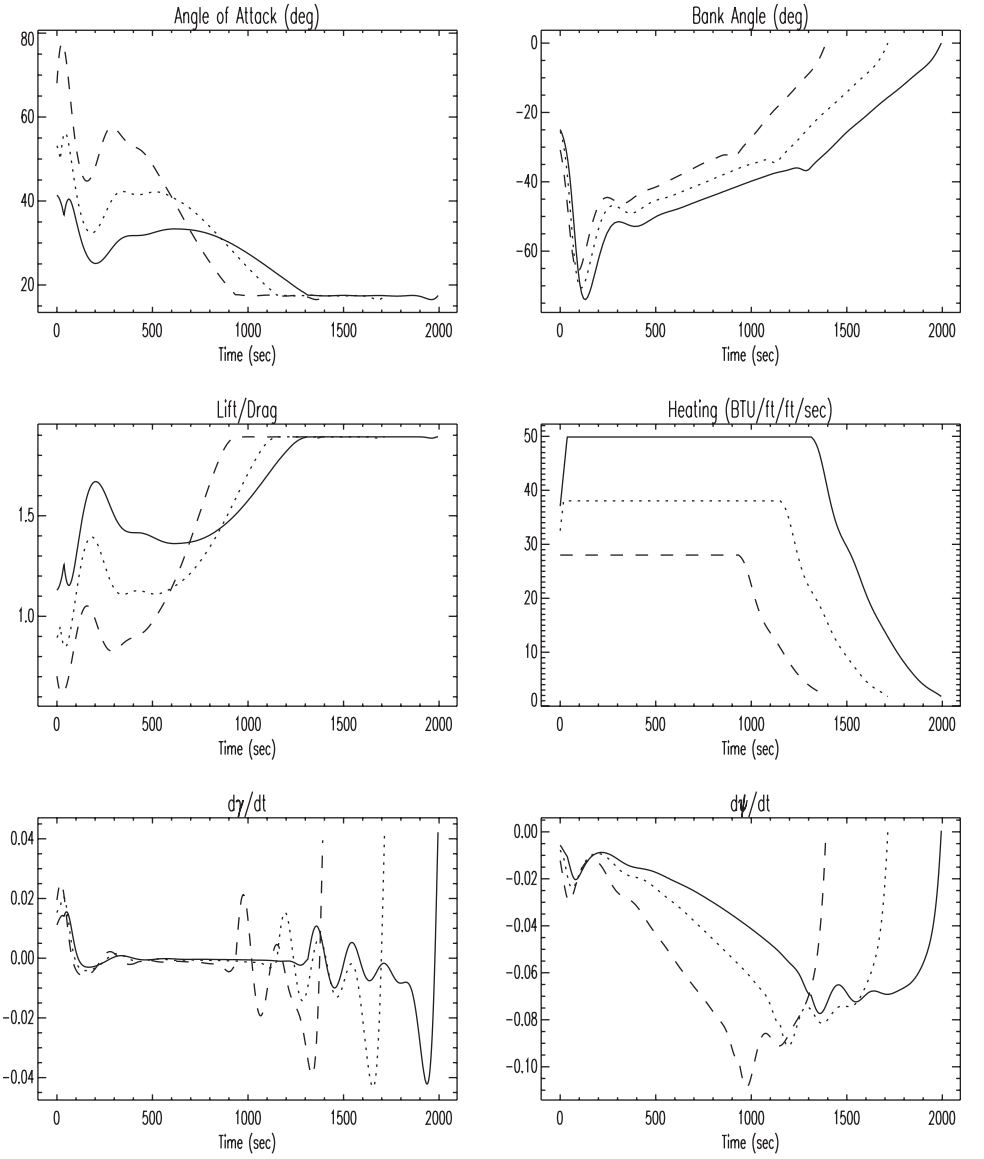


Figure 6.7. *Minimax heating shuttle reentry—control variables, path constraints.*

case, with a solid line used for the $\theta(t_F) = 25$ case, a dotted line used for $\theta(t_F) = 20$, and a dashed line for $\theta(t_F) = 15$. Using the same plot key, Figure 6.7 displays the optimal control and path constraint histories. One additional quantity, the lift to drag ratio L/D, is also displayed. Traditional engineering analysis suggests that aerodynamic efficiency is best when L/D is maximized, and it is readily verified that for the aerodynamics given, the

maximum value of the quantity

$$\frac{L}{D} = \frac{c_L}{c_D}$$

is $(L/D)^* = 1.89211$ which occurs when $\hat{\alpha} = 17.3919$ deg. After a transition period, the final portion of all three optimal trajectories utilizes a "max L/D" angle of attack.

6.2 Minimum Time to Climb

Example 6.4 MINIMUM TIME TO CLIMB. The original minimum time to climb problem was presented by Bryson, Desai, and Hoffman [53] and has been the subject of many analyses since then. Although the problem is not nearly as difficult to solve as the shuttle reentry examples, it is included here because it illustrates the treatment of tabular data. The basic problem is to choose the optimal control function $\alpha(t)$ (the angle of attack) such that an airplane flies from a point on a runway to a specified final altitude as quickly as possible. In its simplest form, the planar motion of the aircraft is described by the following set of ODEs:

$$\dot{h} = v \sin\gamma, \tag{6.14}$$

$$\dot{v} = \frac{1}{m}[T(M,h)\cos\alpha - D] - \frac{\mu}{(R_e+h)^2}\sin\gamma, \tag{6.15}$$

$$\dot{\gamma} = \frac{1}{mv}[T(M,h)\sin\alpha + L] + \cos\gamma\left[\frac{v}{(R_e+h)} - \frac{\mu}{v(R_e+h)^2}\right], \tag{6.16}$$

$$\dot{w} = \frac{-T(M,h)}{I_{sp}}, \tag{6.17}$$

where h is the altitude (ft), v the velocity (ft/sec), γ the flight path angle (rad), w the weight (lb), $m = w/g_0$ the mass, μ the gravitational constant, and R_e the radius of the earth. Furthermore, the simple bounds

$$\begin{aligned} 0 \le h \le 69000 \text{ (ft)}, \qquad & 1 \le v \le 2000 \text{ (ft)}, \\ -89 \text{ (deg)} \le \gamma \le 89 \text{ (deg)}, \qquad & 0 \le w \le 45000 \text{ (lb)}, \\ -20 \text{ (deg)} \le \alpha \le 20 \text{ (deg)} & \end{aligned}$$

are also imposed.

The aerodynamic forces on the vehicle are defined by the expressions

$$D = \frac{1}{2}C_D S\rho v^2, \tag{6.18}$$

$$L = \frac{1}{2}C_L S\rho v^2, \tag{6.19}$$

$$C_L = c_{L\alpha}(M)\alpha, \tag{6.20}$$

$$C_D = c_{D0}(M) + \eta(M)c_{L\alpha}(M)\alpha^2, \tag{6.21}$$

where D is the drag, L is the lift, C_L and C_D are the aerodynamic lift and drag coefficients, respectively, with S the aerodynamic reference area of the vehicle, and ρ is the atmospheric density. Although the results presented here use a cubic spline approximation to the 1962

Standard Atmosphere [46], qualitatively similar results can be achieved with a simple exponential approximation to $\rho(h)$ (cf. Example 6.1). The following constants complete the definition of the problem:

$h(0)$	=	0 (ft),	$h(t_F)$	=	65600.0 (ft),
$v(0)$	=	424.260 (ft/sec),	$v(t_F)$	=	968.148 (ft/sec),
$\gamma(0)$	=	0 (rad),	$\gamma(t_F)$	=	0 (rad),
$w(0)$	=	42000.0 (lb),	S	=	530 (ft^2),
I_{sp}	=	1600.0 (sec),	μ	=	$0.14076539 \times 10^{17}$ (ft^3/sec^2),
g_0	=	32.174 (ft/sec^2),	R_e	=	20902900 (ft).

6.2.1 Tabular Data

As with most real aircraft, the aerodynamic and propulsive forces are specified in tabular form. For the sake of completeness, the data as they appeared in the original reference [53] are given in Tables 6.3 and 6.4. There are a number of significant points that characterize the *tabular data* representation. First, both the thrust and the aerodynamic table values are given to limited numeric precision (i.e., approximately two significant figures). Perhaps the most obvious explanation for the limited precision is that the data probably were originally obtained from experimental tests and truncated at the precision of the test equipment. Unfortunately, the statistical analysis (if any) of the original test data has long since been forgotten. Consequently, it is common to assume that the table values are "exact" and correct to full machine precision. A second difficulty, which is evident in the bivariate thrust data, is that apparently data are missing from the corners of the table. Of course, the data are "missing" because a real aircraft can never fly in these regimes (e.g., at Mach number

Table 6.3. *Propulsion data.*

Thrust $T(M,h)$ (thousands of lb)

	Altitude h (thousands of ft)									
M	0	5	10	15	20	25	30	40	50	70
0.0	24.2									
0.2	28.0	24.6	21.1	18.1	15.2	12.8	10.7			
0.4	28.3	25.2	21.9	18.7	15.9	13.4	11.2	7.3	4.4	
0.6	30.8	27.2	23.8	20.5	17.3	14.7	12.3	8.1	4.9	
0.8	34.5	30.3	26.6	23.2	19.8	16.8	14.1	9.4	5.6	1.1
1.0	37.9	34.3	30.4	26.8	23.3	19.8	16.8	11.2	6.8	1.4
1.2	36.1	38.0	34.9	31.3	27.3	23.6	20.1	13.4	8.3	1.7
1.4		36.6	38.5	36.1	31.6	28.1	24.2	16.2	10.0	2.2
1.6				38.7	35.7	32.0	28.1	19.3	11.9	2.9
1.8						34.6	31.1	21.7	13.3	3.1

Table 6.4. *Aerodynamic data.*

M	0	0.4	0.8	0.9	1.0	1.2	1.4	1.6	1.8
$c_{L\alpha}$	3.44	3.44	3.44	3.58	4.44	3.44	3.01	2.86	2.44
c_{D0}	0.013	0.013	0.013	0.014	0.031	0.041	0.039	0.036	0.035
η	0.54	0.54	0.54	0.75	0.79	0.78	0.89	0.93	0.93

$M = 0$ and $h = 70000$ ft). In fact, for most experimentally obtained data, it can be expected that information will be missing for unrealistic portions of the domain. In view of these realities, it is common to linearly interpolate and never extrapolate the tabular data. Figure 6.8 illustrates a linear treatment of the thrust table.

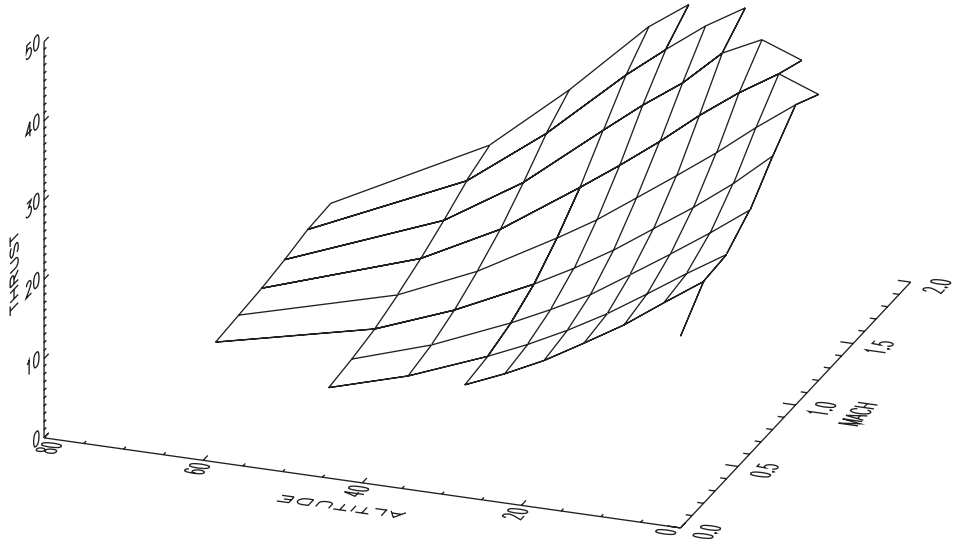


Figure 6.8. *Original thrust data.*

6.2.2 Cubic Spline Interpolation

While a linear treatment of a tabular function may be adequate for simply evaluating the functions, it is totally inappropriate if the functions are to be used within a trajectory simulation and/or optimization. The principle difficulty (as discussed in Section 1.16) stems from the fact that most numerical optimization and integration algorithms assume that the functions are continuously differentiable to at least second order. Thus, just to propagate the trajectory using an integration algorithm such as Runge–Kutta or Adams–Moulton, it is necessary that the right-hand side of the differential equations (6.14)–(6.17) have the necessary continuity. Although it is appealing to ignore a discontinuity, this is usually a poor idea (cf. [106, p. 196]). Similar requirements are imposed when a numerical optimization algorithm is used to shape the trajectory since the optimization uses second derivative (Hessian) information to construct estimates of the solution.

The most direct way to achieve the required continuity is to approximate the data by a tensor product cubic B-spline of the form

$$T(M,h) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} c_{i,j} B_i(M) B_j(h). \tag{6.22}$$

In order to use this approximation, it is necessary to compute the coefficients $c_{i,j}$. The simplest way to compute the spline coefficients is to force the approximating function to

interpolate the data at all of the data points. However, for a unique interpolant, data are required at all points on a rectangular grid. Since data are missing in the corners of the domain, this difficulty is typically resolved by adding "fake" data in the corners using an "eyeball" approach. It is common to ignore the limited precision of the data and simply treat data as though they were of full precision. Finally, by using divided difference estimates of the derivatives at the boundary of the region, the spline coefficients are uniquely determined by solving a sparse system of linear equations. Figures 6.9 and 6.10 illustrate the cubic interpolating spline obtained using this approach.

6.2.3 Minimum Curvature Spline

The cubic spline interpolant does provide C^2 (second derivative) continuity as needed for proper behavior of integration and optimization algorithms. Unfortunately, the approximation, produced by simply interpolating the raw data, does not necessarily reflect the qualitative aspects of the data. In particular, it is common for the interpolant to introduce "wiggles" that are not actually present in the tabular data itself. This is clearly illustrated along the boundaries of the thrust surface in Figure 6.9 and especially in the aerodynamic data for $M \leq 0.8$ as shown in Figure 6.10. In the case of the latter, it is obvious that the interpolant does not reflect the fact that η is constant for low Mach numbers. A second

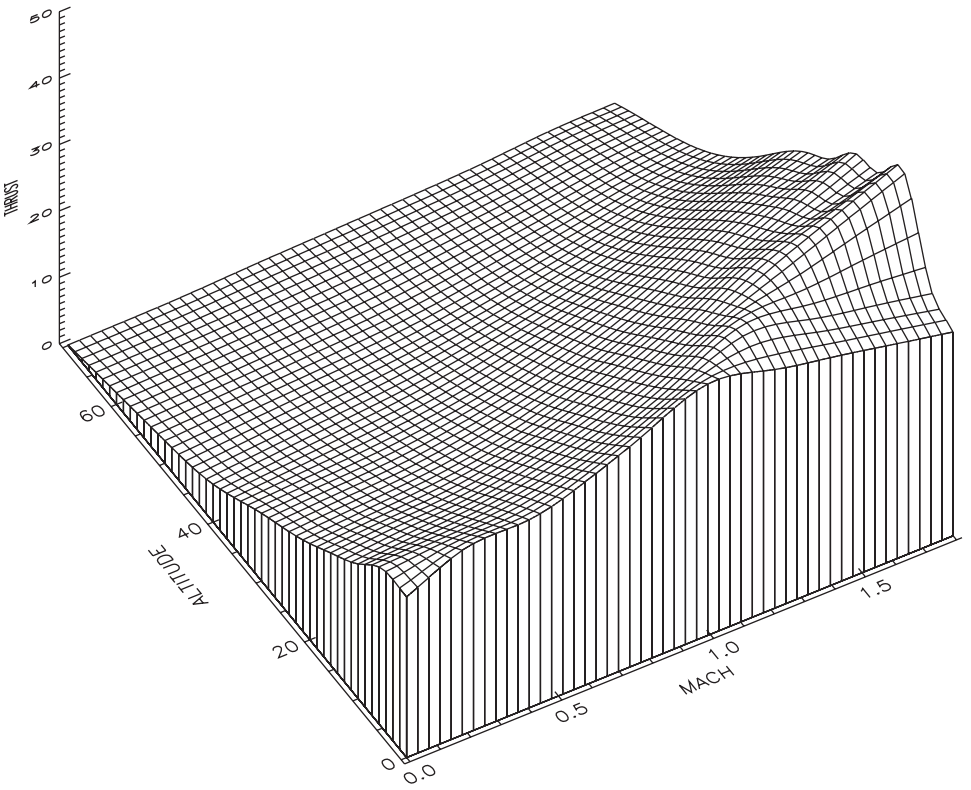


Figure 6.9. *Cubic spline interpolant for thrust data.*

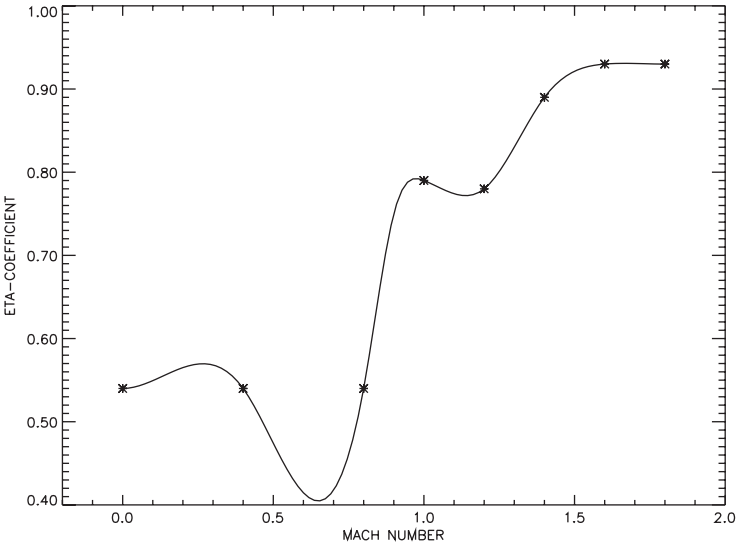


Figure 6.10. *Cubic spline interpolant for aerodynamic data.*

drawback of the cubic interpolant is the need for data at all points on a rectangular grid when constructing an approximation in more than one dimension.

A technique for eliminating the oscillations in the approximation is to carefully select the spline knot locations by inspection of the data, with fewer knots than data points. In this case, it is no longer possible to interpolate the data because the number of coefficients is less than the number of interpolation conditions; i.e., the system is overdetermined. However, it is reasonable to determine the spline coefficients $c_{i,j}$ such that

$$f(c_{i,j}) = \sum_{k=1}^{\ell} \left[T(M_k, h_k) - \hat{T}_k \right]^2 \tag{6.23}$$

is minimized. Furthermore, it is possible to introduce constraints on the slope of the spline approximation to reflect monotonicity of data, i.e.,

$$\left[\frac{\partial T}{\partial M} \right] \geq 0 \qquad \text{if } \left[\hat{T}_{k+1} - \hat{T}_k \right] \geq 0, \tag{6.24}$$

$$\left[\frac{\partial T}{\partial M} \right] \leq 0 \qquad \text{if } \left[\hat{T}_{k+1} - \hat{T}_k \right] \leq 0 \tag{6.25}$$

for $M \in [M_k, M_{k+1}]$. Similar constraints can be imposed to reflect monotonicity in the h-direction. The coefficients that satisfy these conditions can be determined by solving a sparse constrained linear least squares problem using the method described in Section 2.10.

By imposing monotonicity constraints and minimizing the error between the data and the approximating function, it is possible to achieve one of the major goals, namely constructing a C^2 function, which eliminates the wiggles. Unfortunately, the location of the knots in the spline approximation must be chosen such that the coefficients are *well*

determined by minimizing the least square error. In fact, special care must be taken not to locate knots in regions where data are missing since this will result in a rank-deficient least squares problem. In essence, the knots must be located such that local constraint and data uniquely define the spline coefficients.

To resolve these deficiencies, we introduce a rectangular grid with k_1 values in the first coordinate and k_2 values in the second coordinate. We require that all data points lie on the rectangular grid. However, not all grid points need have data (i.e., data can be missing). Then let us consider introducing a spline with (a) double knots at the data points in order to ensure C^2 continuity and (b) single knots at the midpoint of each interval. Then let us determine the spline coefficients $c_{i,j}$ that minimize the "curvature"

$$f(c_{i,j}) = \sum_{k=1}^{L} \left[\frac{\partial^2 T}{\partial M^2}(M_k, h_k) \right]^2 + \left[\frac{\partial^2 T}{\partial h^2}(M_k, h_k) \right]^2 \tag{6.26}$$

and satisfy the data approximation constraints

$$\hat{T}_k - \epsilon \le T(M_k, h_k) \le \hat{T}_k + \epsilon \tag{6.27}$$

for all data points $k = 1, \ldots, \ell$, where ϵ is the data precision. In order to ensure full rank in the Hessian of the objective, we evaluate the curvature at points determined by the knot-interlacing conditions [69]. We retain the slope constraints (6.24) and (6.25) on the approximation in order to reflect the monotonicity of the data. These coefficients can be determined by solving a sparse constrained linear least squares problem. The resulting approximations are illustrated in Figures 6.11 and 6.12. For this particular fit, the least squares problem had 900 variables with 988 constraints of which 77 were equalities. In

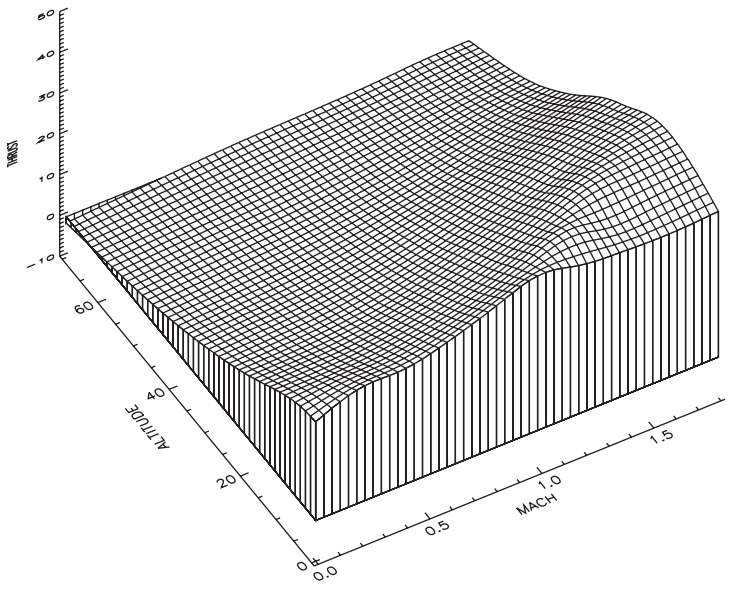


Figure 6.11. *Minimum curvature spline for thrust data.*

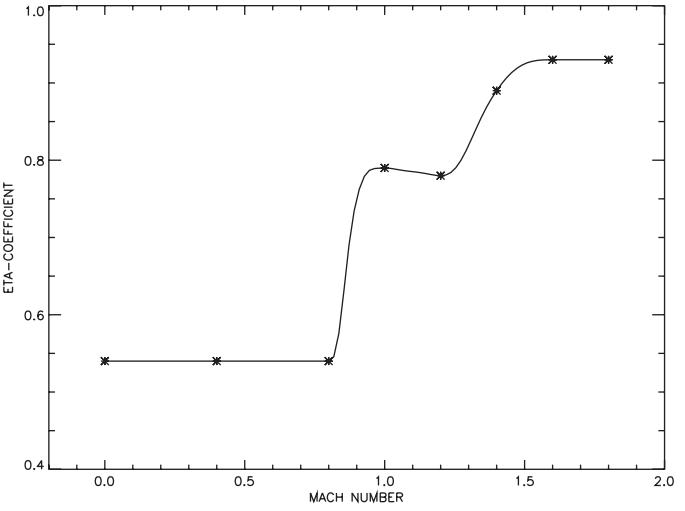


Figure 6.12. *Minimum curvature spline for aerodynamic data.*

general, the number of variables n for a bivariate minimum curvature fit is $n = 9k_1k_2$. For the general case, the minimum curvature formulation requires imposition of m constraints, where $10k_1k_2 \leq m \leq 14k_1k_2$ and the exact number depends on the number of algebraic sign changes in the slope of the data. In addition, if interpolation is required, the number of equality constraints is $m_e \leq k_1k_2$. The entire procedure described for this application has been automated for general multivariate functions with missing data as part of the $\mathbb{SOCS}$ software library.

6.2.4 Numerical Solution

Using the minimum curvature approximations for the tabular data, the minimum time to climb problem can be solved using the direct transcription algorithm in $\mathbb{SOCS}$. Table 6.5 summarizes the progress of the algorithm for this application using a linear initial guess for the dynamic variables. The first grid used a trapezoidal discretization (TR) with 10 grid points. The NLP problem was solved using 25 gradient evaluations (GE), 16 Hessian evaluations (HE), and a total of 523 function evaluations (FE) including the finite difference perturbations. The right-hand sides of the ODEs were evaluated 5230 times (NRHS) leading to a solution with a discretization error of $\epsilon_{\max} = 0.35$. Because the error was not sufficiently equidistributed, a second iteration using the trapezoidal discretization was performed. The HSS discretization (HS) was used for the third, fourth, and fifth refinement iterations, after which the HSC method (HC) was used for the remaining refinements. Figure 6.13 illustrates the progress of the mesh-refinement algorithm with the first refinement iteration shaded darkest and the last refinement shaded lightest. For this case, nine mesh-refinement iterations were required.

Figure 6.14 shows the solution with altitude in multiples of 10000 ft, velocity in multiples of 100 ft/sec, and weight in multiples of 10000 lb. The optimal (minimum) time for this trajectory is 324.9750302 (sec). The altitude time history demonstrates one

Table 6.5. *Minimum time to climb example.*

Iter.	Disc.	M	GE	HE	FE	NRHS	$\epsilon_{\max}$	CPU (sec)
1	TR	10	25	16	523	5230	0.35×10^{0}	2.8
2	TR	19	8	4	159	3021	0.68×10^{-1}	1.7
3	HS	19	8	5	174	6438	0.87×10^{-2}	3.4
4	HS	37	5	1	74	5402	0.51×10^{-3}	3.7
5	HS	59	4	1	61	7137	0.68×10^{-4}	7.5
6	HC	117	4	1	154	35882	0.12×10^{-4}	11.
7	HC	179	4	1	154	54978	0.13×10^{-5}	16.
8	HC	275	4	1	154	84546	0.14×10^{-6}	27.
9	HC	285	3	1	129	73401	0.97×10^{-7}	22.
Total	-	-	65	31	1582	276035	-	94.88

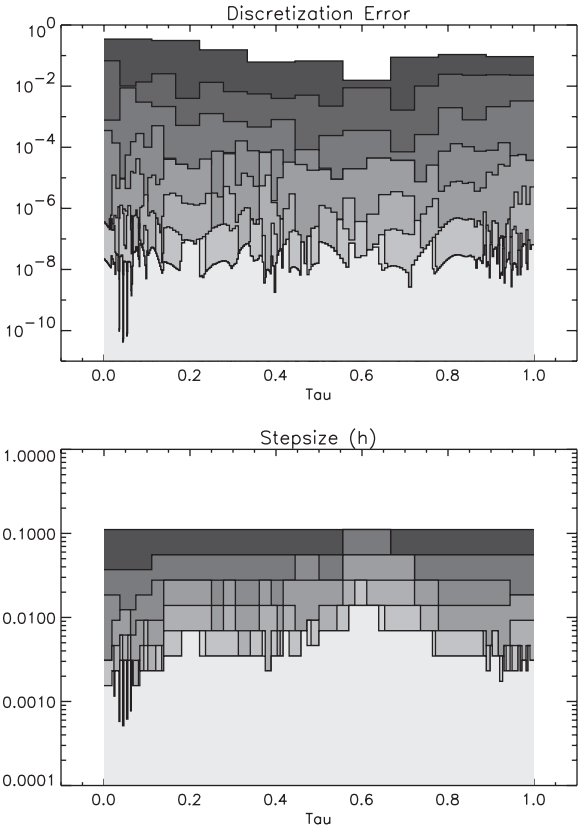


Figure 6.13. *Minimum time to climb—mesh refinement.*

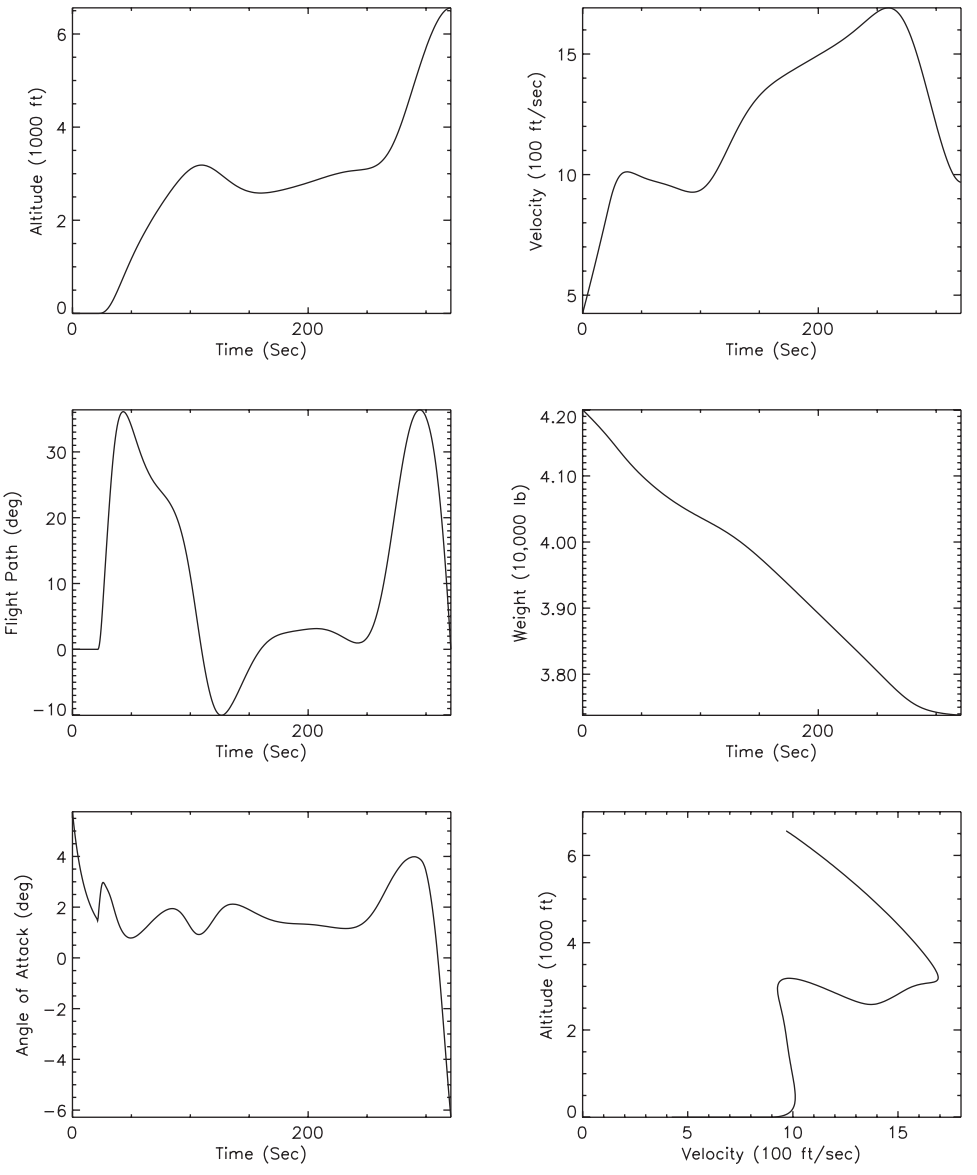


Figure 6.14. *Minimum time to climb solution.*

of the more amazing features of the optimal solution, namely the appearance of a *dive* midway through the minimum time to climb trajectory. When first presented in 1969, this unexpected behavior sparked considerable interest and led to the so-called energy-state approach to trajectory analysis. In particular, along the final portion of the trajectory, the energy is nearly constant, as illustrated in the plot of altitude versus velocity.

6.3 Low-Thrust Orbit Transfer

Example 6.5 LOW-THRUST ORBIT TRANSFER. Constructing the trajectory for a spacecraft as it transfers from a low earth orbit to a mission orbit leads to a class of challenging optimal control examples. The dynamics are very nonlinear and, because the thrust applied to the vehicle is small in comparison to the weight of the spacecraft, the duration of the trajectory can be very long. Problems of this type have been of considerable interest in the aerospace industry [13, 15, 19, 20, 31, 74, 76, 150, 176]. Typically, the goal is to construct the optimal steering during the transfer such that the final weight is maximized (i.e., minimum fuel consumed).

The motion of a vehicle can be described by a system of second order ODEs

$$\ddot{\mathbf{r}} + \mu \frac{\mathbf{r}}{r^3} = \mathbf{a}_d, \tag{6.28}$$

where the radius $r = \|\mathbf{r}\|$ is the magnitude of the inertial position vector $\mathbf{r}$ and μ is the gravitational constant. In this formulation, we define the vector $\mathbf{a}_d$ as the *disturbing acceleration*. This representation for the equations of motion is referred to as the Gauss form of the variational equations.

The Gauss form of the equations of motion isolates the disturbing acceleration from the central force gravitational acceleration. Note that when the disturbing acceleration is zero, $\|\mathbf{a}_d\| = 0$, the fundamental system (6.28) is just a two-body problem. The solution of the two-body problem can, of course, be stated in terms of the constant orbital elements. For low-thrust trajectories, this formulation is appealing because we expect $\|\mathbf{a}_d\|$ to be "small" and, consequently, we expect that the solution can be described in terms of "almost constant" orbital elements. In order to exploit the benefits of the variational form of the differential equations (6.28), it is necessary to transform the Cartesian state into an appropriate set of orbit elements. One potential set contains the classical elements $(a, e, i, \Omega, \omega, M)$. However, these elements exhibit singularities for $e = 0$ and $i = 0$ deg or 90 deg. A set of *equinoctial* orbital elements that avoid the singularities in the classical elements has been described in [6], [49], and [74]. Kechichian developed a particular form of these equations in [122], [121], and [174]. These equations were used to solve a low-thrust earth orbit transfer problem as described in [13]. Unfortunately, this set of equinoctial elements does not accommodate orbits with $e \geq 1$. To eliminate this deficiency, a modified set of equinoctial orbit elements is described in [15] based on the work in [171].

6.3.1 Modified Equinoctial Coordinates

The dynamics of the system can be described in terms of the state variables

$$[\mathbf{y}^\mathsf{T}, w] = [p, f, g, h, k, L, w], \tag{6.29}$$

the control variables

$$\mathbf{u}^\mathsf{T} = [u_r, u_\theta, u_h], \tag{6.30}$$

and the unknown parameter τ.

Using the modified equinoctial elements, the equations of motion for a vehicle with variable thrust can be stated as

$$\dot{\mathbf{y}} = \mathbf{A}(\mathbf{y})\boldsymbol{\Delta} + \mathbf{b}, \tag{6.31}$$

$$\dot{w} = -T\,[1 + 0.01\tau]/I_{sp}, \tag{6.32}$$

$$0 = \|\mathbf{u}\| - 1, \tag{6.33}$$

$$\tau_L \quad \leq \tau \leq \quad 0. \tag{6.34}$$

The equinoctial dynamics are defined by the matrix

$$\mathbf{A} = \begin{bmatrix} 0 & \frac{2p}{q}\sqrt{\frac{p}{\mu}} & 0 \\ \sqrt{\frac{p}{\mu}}\sin L & \sqrt{\frac{p}{\mu}}\frac{1}{q}\{(q+1)\cos L + f\} & -\sqrt{\frac{p}{\mu}}\frac{g}{q}\{h\sin L - k\cos L\} \\ -\sqrt{\frac{p}{\mu}}\cos L & \sqrt{\frac{p}{\mu}}\frac{1}{q}\{(q+1)\sin L + g\} & \sqrt{\frac{p}{\mu}}\frac{f}{q}\{h\sin L - k\cos L\} \\ 0 & 0 & \sqrt{\frac{p}{\mu}}\frac{s^2\cos L}{2q} \\ 0 & 0 & \sqrt{\frac{p}{\mu}}\frac{s^2\sin L}{2q} \\ 0 & 0 & \sqrt{\frac{p}{\mu}}\frac{1}{q}\{h\sin L - k\cos L\} \end{bmatrix} \tag{6.35}$$

and the vector

$$\mathbf{b}^{\mathsf{T}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \sqrt{\mu p}\left(\frac{q}{p}\right)^2 \end{bmatrix}, \tag{6.36}$$

where

$$q = 1 + f\cos L + g\sin L, \tag{6.37}$$

$$r = \frac{p}{q}, \tag{6.38}$$

$$\alpha^2 = h^2 - k^2, \tag{6.39}$$

$$\chi = \sqrt{h^2 + k^2}, \tag{6.40}$$

$$s^2 = 1 + \chi^2. \tag{6.41}$$

The equinoctial coordinates $\mathbf{y}$ are related to the Cartesian state $(\mathbf{r}, \mathbf{v})$ according to the expressions

$$\mathbf{r} = \begin{bmatrix} \frac{r}{s^2}\left(\cos L + \alpha^2\cos L + 2hk\sin L\right) \\ \frac{r}{s^2}\left(\sin L - \alpha^2\sin L + 2hk\cos L\right) \\ \frac{2r}{s^2}\left(h\sin L - k\cos L\right) \end{bmatrix}, \tag{6.42}$$

$$\mathbf{v} = \begin{bmatrix} -\frac{1}{s^2}\sqrt{\frac{\mu}{p}}\left(\sin L + \alpha^2\sin L - 2hk\cos L + g - 2fhk + \alpha^2 g\right) \\ -\frac{1}{s^2}\sqrt{\frac{\mu}{p}}\left(-\cos L + \alpha^2\cos L + 2hk\sin L - f + 2ghk + \alpha^2 f\right) \\ \frac{2}{s^2}\sqrt{\frac{\mu}{p}}\left(h\cos L + k\sin L + fh + gk\right) \end{bmatrix}. \tag{6.43}$$

As a result of this transformation, the disturbing acceleration vector $\mathbf{a}_d$ in (6.28) is replaced by

$$\boldsymbol{\Delta} = \boldsymbol{\Delta}_g + \boldsymbol{\Delta}_T \tag{6.44}$$

with a contribution due to oblate earth effects $\boldsymbol{\Delta}_g$ and another caused by thrust $\boldsymbol{\Delta}_T$. The disturbing acceleration is expressed in a rotating radial frame whose principle axes are defined by

$$\mathbf{Q}_r = \begin{bmatrix} \mathbf{i}_r & \mathbf{i}_\theta & \mathbf{i}_h \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{r}}{\|\mathbf{r}\|} & \frac{(\mathbf{r}\times\mathbf{v})\times\mathbf{r}}{\|\mathbf{r}\times\mathbf{v}\|\|\mathbf{r}\|} & \frac{\mathbf{r}\times\mathbf{v}}{\|\mathbf{r}\times\mathbf{v}\|} \end{bmatrix}. \tag{6.45}$$

As stated, (6.31)–(6.34) are perfectly general and describe the motion of a point mass when subject to the disturbing acceleration vector $\boldsymbol{\Delta}$. Notice that when the disturbing acceleration is zero, $\boldsymbol{\Delta} = 0$, the first five equations are simply $\dot{p} = \dot{f} = \dot{g} = \dot{h} = \dot{k} = 0$, which implies that the elements are constant. It is important to note that the disturbing acceleration vector can be attributed to any perturbing force(s). A more complete derivation of the equinoctial dynamics can be found in [15].

6.3.2 Gravitational Disturbing Acceleration

Oblate gravity models are typically defined in a local horizontal reference frame, that is,

$$\boldsymbol{\delta}\mathbf{g} = \delta g_n \mathbf{i}_n - \delta g_r \mathbf{i}_r, \tag{6.46}$$

where

$$\mathbf{i}_n = \frac{\mathbf{e}_n - (\mathbf{e}_n^\top \mathbf{i}_r)\mathbf{i}_r}{\|\mathbf{e}_n - (\mathbf{e}_n^\top \mathbf{i}_r)\mathbf{i}_r\|} \tag{6.47}$$

defines the local north direction with $\mathbf{e}_n = (0,0,1)$. A reasonably accurate model is obtained if the tesseral harmonics are ignored and only the first four zonal harmonics are included in the geopotential function. In this case, the oblate earth perturbations to the gravitational acceleration are given by

$$\delta g_n = -\frac{\mu\cos\phi}{r^2}\sum_{k=2}^{4}\left(\frac{R_e}{r}\right)^k P_k' J_k, \tag{6.48}$$

$$\delta g_r = -\frac{\mu}{r^2}\sum_{k=2}^{4}(k+1)\left(\frac{R_e}{r}\right)^k P_k J_k, \tag{6.49}$$

where ϕ is the geocentric latitude, R_e is the equatorial radius of the earth, $P_k(\sin\phi)$ is the kth-order Legendre polynomial with corresponding derivative P_k', and J_k are the zonal harmonic coefficients. Finally, to obtain the gravitational perturbations in the rotating radial frame, it follows that

$$\boldsymbol{\Delta}_g = \mathbf{Q}_r^\top \boldsymbol{\delta}\mathbf{g}. \tag{6.50}$$

6.3.3 Thrust Acceleration—Burn Arcs

To this point, the discussion has concentrated on incorporating perturbing forces due to oblate earth effects. Of course, the second major perturbation is the thrust acceleration defined by

$$\boldsymbol{\Delta}_T = \frac{g_o T\,[1+.01\tau]}{w}\mathbf{u}, \tag{6.51}$$

where T is the maximum thrust and $\tau_L \leq \tau \leq 0$ is a throttle factor. In general, the direction of the thrust acceleration vector, which is defined by the time-varying control vector $\mathbf{u}(t) = (u_r, u_\theta, u_h)$, can be chosen arbitrarily as long as the vector has unit length at all points in time. This is achieved using the path constraint (6.33). The magnitude of the thrust is, of course, related to the vehicle weight according to (6.32), where g_0 is the mass to weight conversion factor and the specific impulse of the motor is denoted by I_{sp}. Defining the thrust direction using the vector $\mathbf{u}(t)$ and path constraint $\|\mathbf{u}(t)\| = 1$ is particularly well suited for missions that involve steering over large portions of the trajectory, as illustrated in [13], because ambiguities in the pointing direction are avoided. Specifying the thrust direction by two angles (e.g., yaw and pitch), which are treated as control variables, is not unique since the angles $\alpha = \alpha_0 \pm 2k\pi$ all yield the *same* direction. In contrast, there is a *unique* set of control variables $\mathbf{u}$ corresponding to any thrust direction. This ambiguity is demonstrated in Figure 6.15.

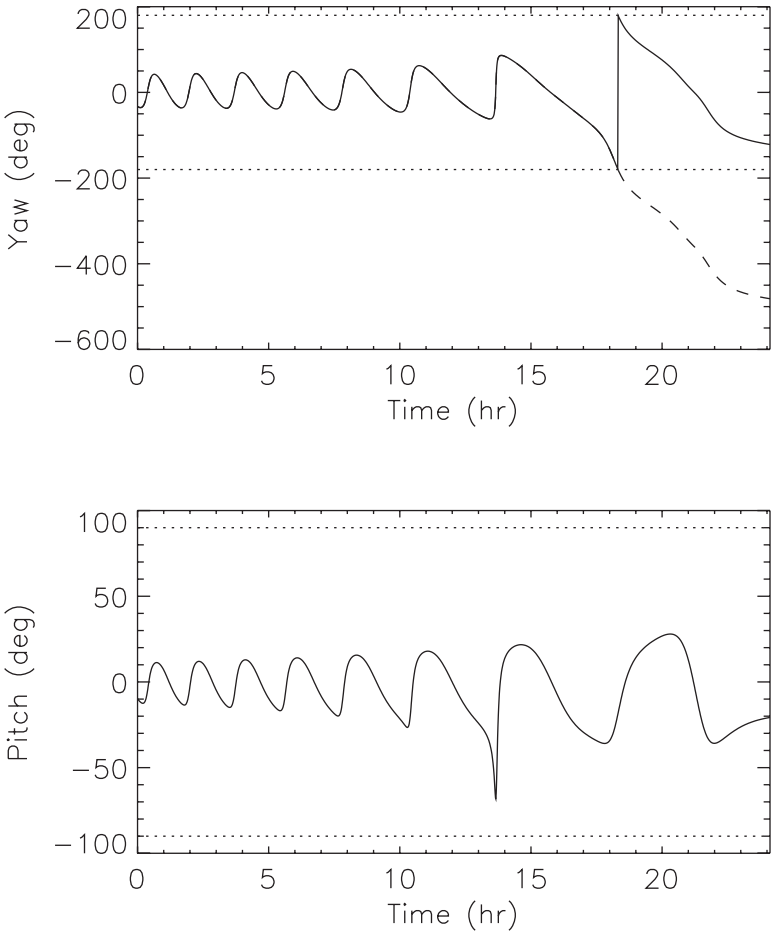


Figure 6.15. *Optimal control angles.*

6.3.4 Boundary Conditions

The standard approach for defining the boundary conditions of an orbit transfer problem is to specify the final state in terms of the instantaneous or *osculating* orbit elements at the burnout time t_F. The final orbit for this example has an apogee altitude of 21450 nm, a perigee altitude of 350 nm, an inclination of 63.4 deg, and an argument of perigee of 270 deg. This final orbit can be defined by the boundary conditions (all evaluated at $t = t_F$)

$$p = 40007346.015232 \text{ (ft)}, \tag{6.52}$$

$$\sqrt{f^2 + g^2} = 0.73550320568829, \tag{6.53}$$

$$\sqrt{h^2 + k^2} = 0.61761258786099, \tag{6.54}$$

$$fh + gk = 0, \tag{6.55}$$

$$gh - kf \leq 0. \tag{6.56}$$

The initial orbit for this example is a "standard" space shuttle park orbit, which is circular at an altitude of 150 nm, and an inclination of 28.5 deg. This particular orbit leads to the following values for the equinoctial elements at $t = 0$:

$$p = 21837080.052835 \text{ (ft)}, \quad f = 0, \quad g = 0,$$
$$h = -0.25396764647494, \quad k = 0, \quad L = \pi \text{ (rad)}.$$

The following constants complete the definition of the problem:

$$w(0) = 1 \text{ (lb)}, \qquad g_0 = 32.174 \text{ (ft/sec}^2\text{)},$$
$$I_{sp} = 450 \text{ (sec)}, \qquad T = 4.446618 \times 10^{-3} \text{ (lb)},$$
$$\mu = 1.407645794 \times 10^{16} \text{ (ft}^3\text{/sec}^2\text{)}, \qquad R_e = 20925662.73 \text{ (ft)},$$
$$J_2 = 1082.639 \times 10^{-6}, \qquad J_3 = -2.565 \times 10^{-6},$$
$$J_4 = -1.608 \times 10^{-6}, \qquad \tau_L = -50.$$

For convenience, we have chosen the initial weight as 1 lb, and the goal is to maximize the final weight, i.e., $w(t_F)$.

6.3.5 Numerical Solution

The solution to this low-thrust orbit transfer problem obtained using the implementation in $\mathbb{SOCS}$ is summarized in Figures 6.16 and 6.17. All times are plotted in hours, with the semiparameter p given in millions of feet and the true longitude L in multiples of 360 deg. The iteration history is summarized in Table 6.6. Figure 6.18 illustrates the behavior of the mesh-refinement algorithm. The optimal value for the final weight is $w^*(t_F) = 0.2201791266$ lb, which occurs at $t_F^* = 86810.0$ sec with an optimal throttle parameter value of $\tau^* = -9.09081$. The final trajectory profile is illustrated in Figure 6.19. It is also worthwhile to examine the behavior of the optimal control history when angles are used instead of the direction vector $\mathbf{u}(t)$. In particular, one can define

$$u_r = -\sin\theta,$$
$$u_\theta = \cos\theta\cos\psi,$$
$$u_h = -\cos\theta\sin\psi$$

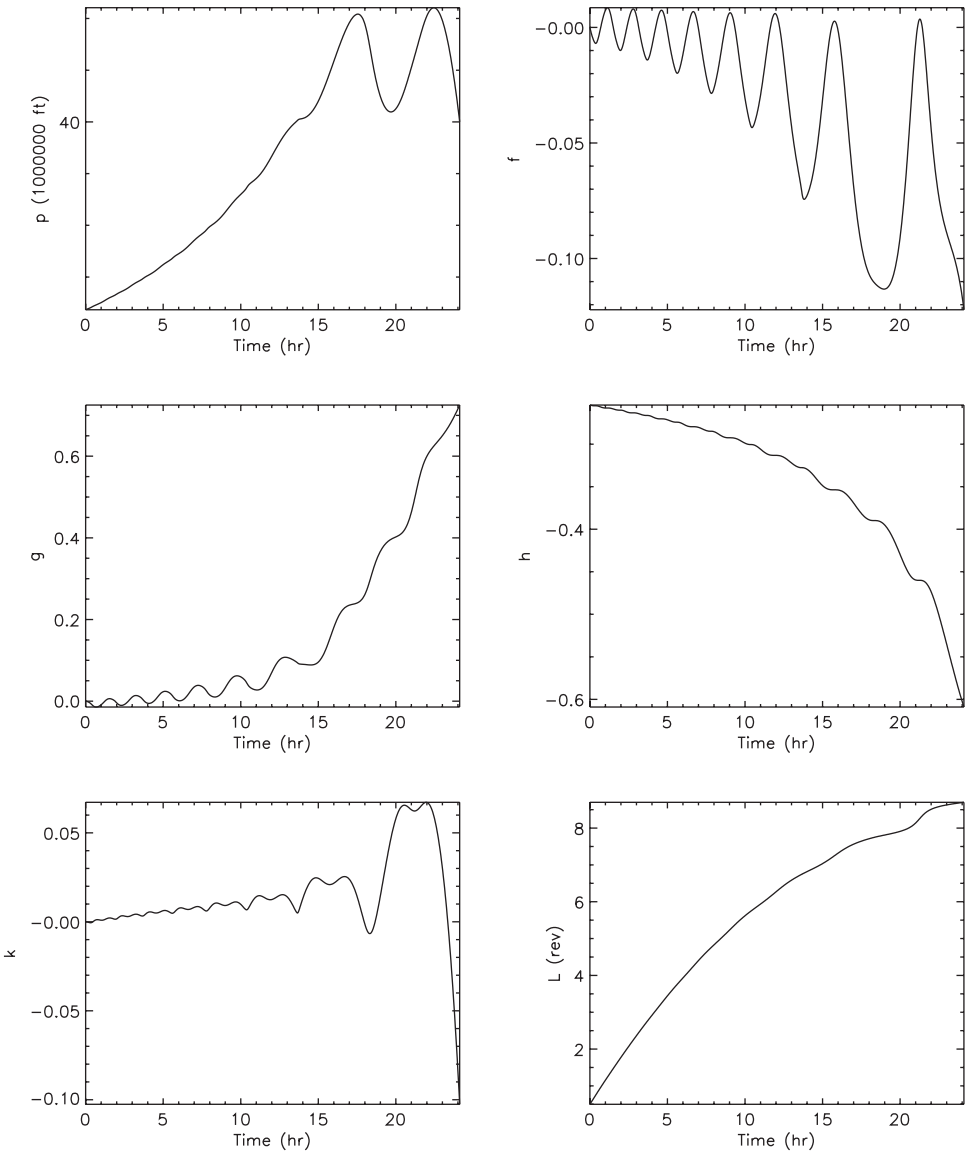


Figure 6.16. *Low-thrust transfer—state variables.*

using the two control angles (θ, ψ). The time history of these angles illustrated in Figure 6.15 clearly demonstrates the ambiguity in the yaw angle. In fact, if the controls were modeled using angles, it is clear that the mesh-refinement procedure would detect an apparent discontinuity caused by the yaw angle "wrapping" unless the dotted time history was followed.

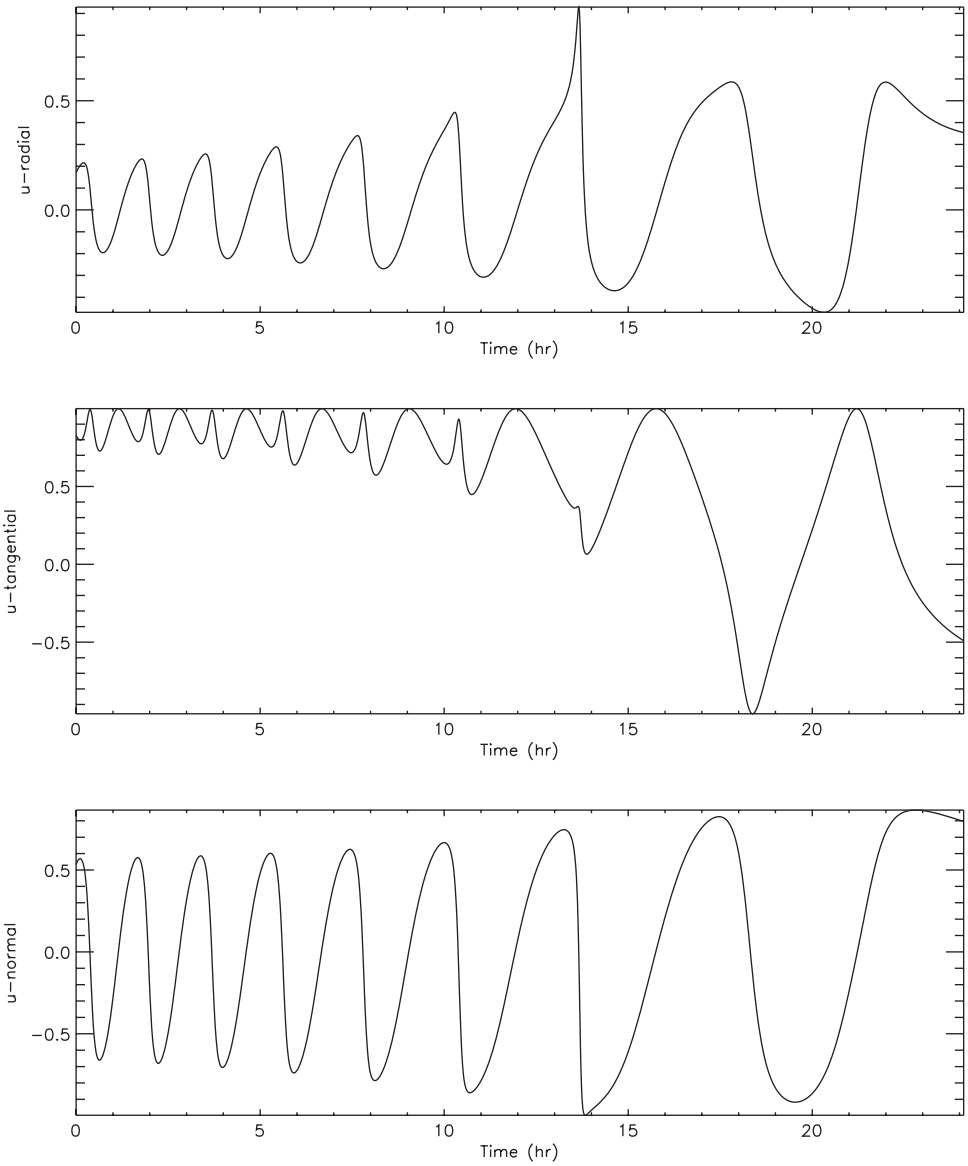


Figure 6.17. *Low-thrust transfer—control variables.*

6.4 Two-Burn Orbit Transfer

In this section, consider a problem similar to Example 6.5, with one important difference—the magnitude of the thrust. In particular, for this example the thrust $T = 1.25$ lb and the initial weight $w(0) = 1$ lb leading to a description that is very representative of most

Table 6.6. *Low-thrust transfer example.*

Iter.	Disc.	M	GE	HE	FE	RHS	$\epsilon_{\max}$	CPU (sec)
1	TR	150	416	80	11231	1684650	$.22 \times 10^{-2}$	0.23×10^{4}
2	TR	299	10	7	604	180596	$.26 \times 10^{-3}$	0.92×10^{2}
3	HS	299	11	8	682	407154	$.31 \times 10^{-4}$	0.31×10^{3}
4	HS	597	8	6	503	600079	$.15 \times 10^{-5}$	0.68×10^{3}
5	HS	966	6	3	292	563852	$.61 \times 10^{-7}$	0.12×10^{4}
Total	-	-	451	104	13312	3436331	-	4562.20

operational launch vehicles. To make the problem more concrete, let us suppose that the vehicle begins in the same standard space shuttle park orbit as in Example 6.5, which is a 150 nm circular orbit with an inclination of 28.5 deg. The final orbit for this example is chosen to be a circular orbit with an altitude of 19323 nm and an inclination of 0 deg. This orbit is referred to as *geosynchronous* because it has a period of 24 hr, and the motion of a vehicle in this orbit is synchronized with the revolution of the earth beneath it. Thus, a satellite placed in geosynchronous orbit appears motionless when viewed from the earth, making it an attractive platform for communication systems. Since most communication satellites are placed in an orbit of this type, this particular trajectory design problem has been studied extensively, and, in contrast to Example 6.5, which is a hard problem, this may be considered an easy problem. On the other hand, because the problem is rather simple to solve, it permits us to illustrate and compare different solution methods.

The vehicle dynamics are initialized at a point in the park orbit. For convenience, the point at which the vehicle crosses the equator in a northbound direction (referred to as the ascending node) is chosen to be the initial time. After coasting for an unspecified time, the vehicle's engine is ignited. The orientation and duration of this "first burn" are also unspecified. The additional velocity added by the first burn places the vehicle into a "transfer orbit." After coasting for an unspecified time in the transfer orbit, the motor is again ignited and the "second burn" is performed. The duration of the burn and orientation of the vehicle during this time are also unspecified. However, when the second burn is completed, the vehicle must be deployed in the desired geosynchronous orbit. There are a number of ways to quantify an optimal orbit transfer. Typically, the trajectory that minimizes the fuel consumed or maximizes the final weight is preferred. An equivalent approach is to design a trajectory that minimizes the energy added by the propulsion system, and this approach is referred to as a "minimum Δv" transfer.

The motion of a vehicle can be described by the following system of first order ODEs:

$$\dot{\mathbf{r}} = \mathbf{v}, \tag{6.57}$$

$$\dot{\mathbf{v}} = \mathbf{g} + \mathbf{T}, \tag{6.58}$$

$$\dot{w} = -T/I_{sp}. \tag{6.59}$$

These equations are equivalent to (6.28), where $\mathbf{r}$ is the inertial position vector, $\mathbf{v}$ is the inertial velocity vector, and w is the weight. The gravitational acceleration is defined by $\mathbf{g}$ and the thrust acceleration is defined by $\mathbf{T}$. We use T to denote the magnitude of the thrust $\|\mathbf{T}\|$ and I_{sp} to denote the specific impulse.

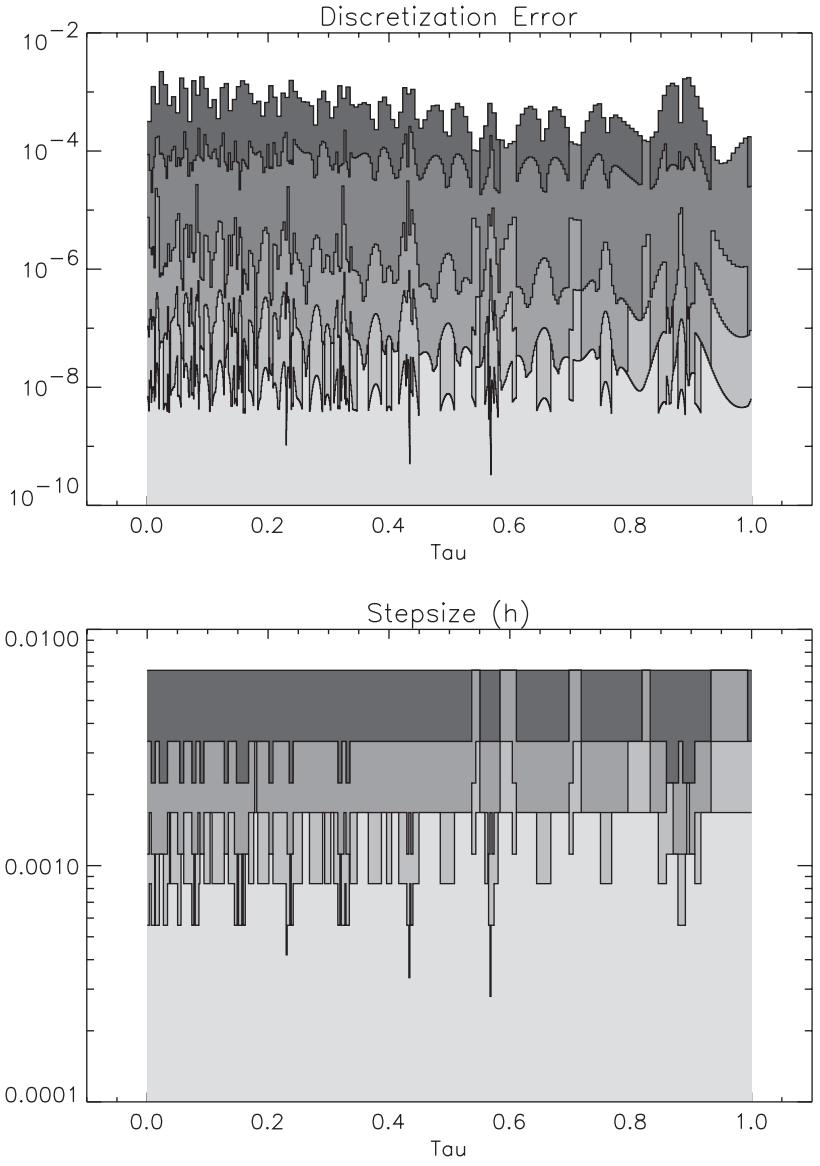


Figure 6.18. *Low-thrust transfer—mesh refinement.*

6.4.1 Simple Shooting Formulation

Example 6.6 TWO-BURN TRANSFER. In Section 3.3, we described the *shooting method*, which is often used for solving simple BVPs such as this two-burn transfer. Early attempts to solve this problem introduced approximations to the *physics* in order to expedite the solution of (6.57)–(6.59). Although a simple shooting formulation can be obtained without approximating the physics, we will follow the historical technique.

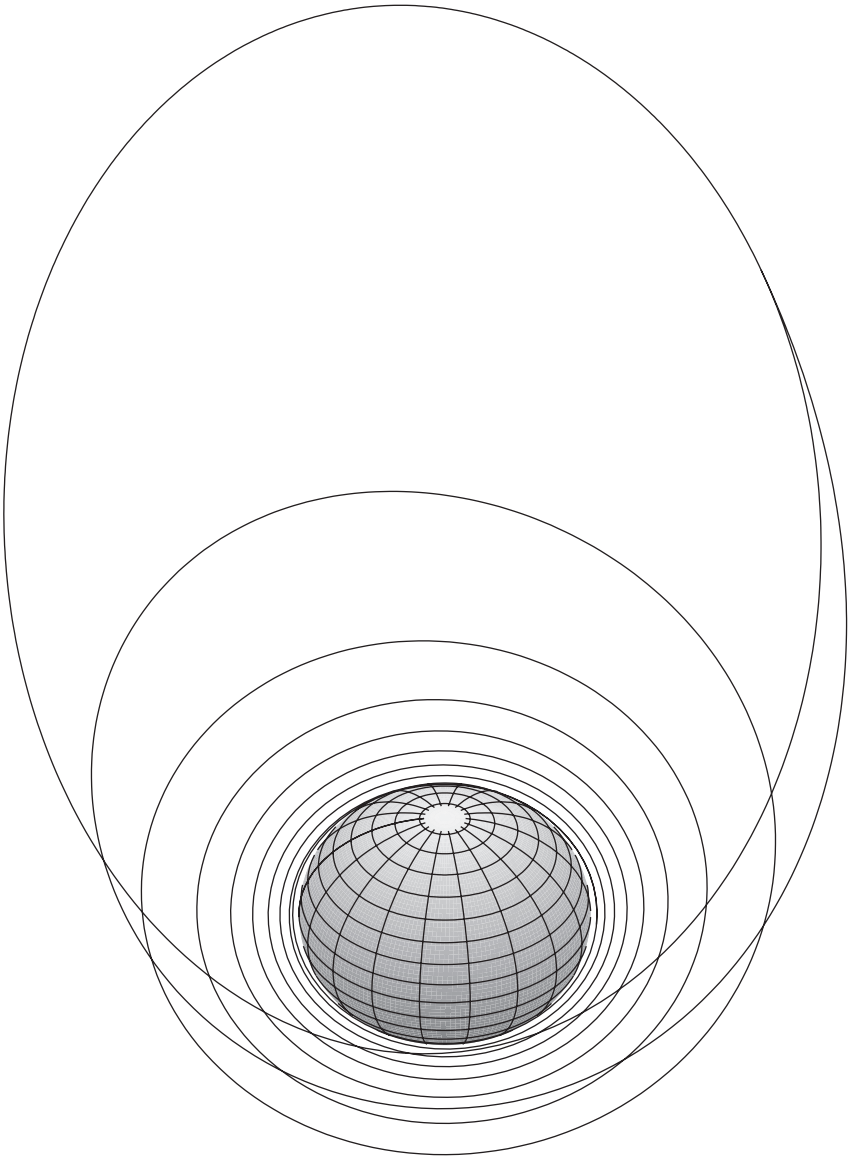


Figure 6.19. *Optimal low-thrust transfer.*

When the thrust T is large, the duration of the burns is very short compared to the overall transfer time. Thus, it is common to assume the burns occur instantaneously. Using this approximation, the net effect of a burn is to change the *velocity* but not the *position*. This technique is called an *impulsive* Δv approximation. To be more precise,

$$\mathbf{v}(t_2) = \mathbf{v}(t_1) + \boldsymbol{\Delta v}, \tag{6.60}$$

where $\mathbf{v}(t_1)$ is the velocity before the burn, $\mathbf{v}(t_2)$ is the velocity after the burn, $\boldsymbol{\Delta v}$ is the velocity added by the burn, and $t_1 = t_2$. The velocity change is related to the weight by the expression

$$\|\boldsymbol{\Delta v}\| = g_0 I_{sp} \ln\left[\frac{w(t_1)}{w(t_2)}\right], \tag{6.61}$$

where $g_0 = 32.174$ (ft/sec^2) is the mass to weight conversion factor. Also, for convenience, we can define the impulsive velocity using spherical coordinates $(\Delta v, \theta, \psi)$, where

$$\boldsymbol{\Delta v} = \mathbf{Q}_v \begin{pmatrix} \Delta v \cos\theta \cos\psi \\ \Delta v \cos\theta \sin\psi \\ \Delta v \sin\theta \end{pmatrix} \tag{6.62}$$

and the orthogonal matrix

$$\mathbf{Q}_v = \left[\begin{array}{ccc} \frac{\mathbf{v}}{\|\mathbf{v}\|} & \frac{\mathbf{v}\times\mathbf{r}}{\|\mathbf{v}\times\mathbf{r}\|} & \frac{\mathbf{v}}{\|\mathbf{v}\|} \times \left(\frac{\mathbf{v}\times\mathbf{r}}{\|\mathbf{v}\times\mathbf{r}\|}\right) \end{array} \right] \tag{6.63}$$

defines the principle axes of an inertial velocity coordinate frame.

During coast portions of the trajectory, $\mathbf{T} = \mathbf{0}$ and the equations of motion (6.57)–(6.59) are just

$$\dot{\mathbf{r}} = \mathbf{v}, \tag{6.64}$$

$$\dot{\mathbf{v}} = \mathbf{g}. \tag{6.65}$$

The only nonlinear quantity in these equations is the gravitational acceleration $\mathbf{g}(\mathbf{r})$. If one assumes that the earth is spherical, then $\mathbf{g}(\mathbf{r}) = -\mu\mathbf{r}/r^3$ and the differential equations (6.64)–(6.65) have an analytic solution! Thus, given the state at some time t_0, one can analytically propagate to another time t_1, i.e.,

$$\begin{bmatrix} \mathbf{r}(t_0) \\ \mathbf{v}(t_0) \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{r}(t_1) \\ \mathbf{v}(t_1) \end{bmatrix}. \tag{6.66}$$

In effect, this propagation defines a nonlinear mapping between the states at t_0 and t_1, say

$$\begin{bmatrix} \mathbf{r}(t_1) \\ \mathbf{v}(t_1) \end{bmatrix} = \mathbf{P}\left[\mathbf{r}(t_0), \mathbf{v}(t_0), t_1, t_0\right]. \tag{6.67}$$

Kepler provided the original solution of this propagation problem over 100 years ago, and a complete description of the computational procedure can be found both in Section 7.1.4 and [78]. More recently, Huffman [115] has extended the analytic propagation to include the most significant oblate earth perturbation (i.e., including the contribution of J_2 in (6.48) and (6.49)).

At this point, it is worthwhile to expand on a subtle but important detail regarding the meaning of analytic propagation. In order to propagate from one *time* to another, it is necessary to solve Kepler's equation. In its simplest form, one must compute E such that

$$n(t - t_0) = E - e\sin E, \tag{6.68}$$

where n and e are constants for the orbit (the mean motion and eccentricity) and t is the propagation time. Typically, this transcendental equation is solved by a Newton iteration.

The variable E, called the eccentric anomaly, determines the angular position of the vehicle at time t on the orbit. In fact, for orbital motion, there is an angular change α that will satisfy the equation

$$k[\alpha, \Delta t] = 0 \tag{6.69}$$

for a specified value of Δt, but it cannot be written explicitly in the form

$$\alpha \sim k^{-1}[\Delta t].$$

Thus, if we choose to propagate the orbit by specifying the time change, we must solve a transcendental equation. But an "internal" iteration is undesirable, as discussed in Section 1.16. On the other hand, if we choose to propagate by specifying an angular change, no iteration is required! Thus, the proper formulation is to introduce the angular change α as an NLP variable (in addition to t) and then impose the Kepler equation (6.69). Section 7.1.4 revisits this issue.

The fundamental idea of combining impulsive Δv and analytic orbit propagation was originally proposed by a German engineer, Walter Hohmann, who published the idea (in German) in Munich in 1925. Although his original paper described transfers between circular orbits with the same inclination, the term "Hohmann transfer" is now loosely applied to describe any transfer between arbitrary orbits using impulsive approximations.

The boundary conditions that define the desired geosynchronous orbit are

$$(\|\mathbf{r}\| - R_e)/\sigma = 19323 \text{ (nm)}, \tag{6.70}$$

$$\|\mathbf{v}\| = 10087.5 \text{ (ft/sec)}, \tag{6.71}$$

$$\frac{\mathbf{v}^{\mathsf{T}}\mathbf{r}}{\|\mathbf{v}\|\|\mathbf{r}\|} = 0, \tag{6.72}$$

$$\frac{v_3}{\|\mathbf{v}\|} = 0, \tag{6.73}$$

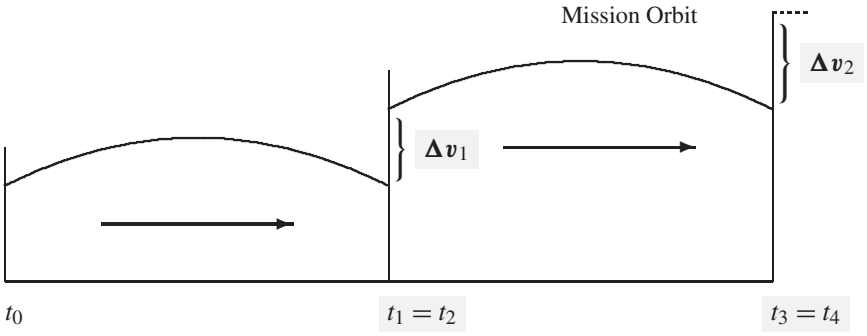
$$\frac{r_3}{\|\mathbf{r}\|} = 0, \tag{6.74}$$

where the constant $\sigma = 6076.1154855643$ ft/nm. We use the same values for R_e, μ, and J_2 as in Example 6.5.

We can now define the NLP problem that must be solved. The NLP variables are

$$\mathbf{x} = \begin{bmatrix} t_1 \\ \alpha_1 \\ \Delta v_1 \\ \theta_1 \\ \psi_1 \\ t_3 \\ \alpha_2 \\ \Delta v_2 \\ \theta_2 \\ \psi_2 \end{bmatrix} = \begin{bmatrix} \text{First-burn ignition time} \\ \text{First coast angle} \\ \text{First-burn velocity} \\ \text{First-burn pitch angle} \\ \text{First-burn yaw angle} \\ \text{Second-burn ignition time} \\ \text{Second coast angle} \\ \text{Second-burn velocity} \\ \text{Second-burn pitch angle} \\ \text{Second-burn yaw angle} \end{bmatrix}. \tag{6.75}$$

When values are specified for the NLP variables, it is possible to compute the objective and constraint functions. This is illustrated schematically below:



For this direct shooting formulation, the function generator is as follows:

Direct Shooting

Input: $\mathbf{x}$
Initialize State:
  $\mathbf{r}(t_0), \mathbf{v}(t_0), t_0$
First Coast:
  Propagate through angle α_1; from (6.67) compute $\mathbf{r}(t_1), \mathbf{v}(t_1)$.
Constraint Evaluation:
  Compute Kepler constraint from (6.69) using α_1, t_1.
First Burn: Given $\Delta v_1, \theta_1, \psi_1$, set $\mathbf{r}(t_2) = \mathbf{r}(t_1)$, $t_2 = t_1$, and compute
  $\mathbf{v}(t_2)$ from (6.60), (6.62), and (6.63).
Second Coast:
  Propagate through angle α_2; from (6.67) compute $\mathbf{r}(t_3), \mathbf{v}(t_3), t_3$.
Constraint Evaluation:
  Compute Kepler constraint from (6.69) using α_2, t_3.
Second Burn: Given $\Delta v_2, \theta_2, \psi_2$, set $\mathbf{r}(t_4) = \mathbf{r}(t_3)$, $t_4 = t_3$, and compute
  $\mathbf{v}(t_4)$ from (6.60), (6.62), and (6.63).
Constraint Evaluation:
  Compute boundary conditions at t_4 from (6.70)–(6.74).
Terminate Trajectory
  Compute objective $F(\mathbf{x}), \mathbf{c}(\mathbf{x})$.
Output: $F(\mathbf{x}), \mathbf{c}(\mathbf{x})$

The goal is to minimize the objective function

$$F(\mathbf{x}) = \Delta v_1 + \Delta v_2. \tag{6.76}$$

Since none of the computed quantities for this example explicitly depends on time, this problem can also be formulated using only propagation angles. Table 6.7 presents the solutions obtained with and without time as a propagation variable (formulations A and B,

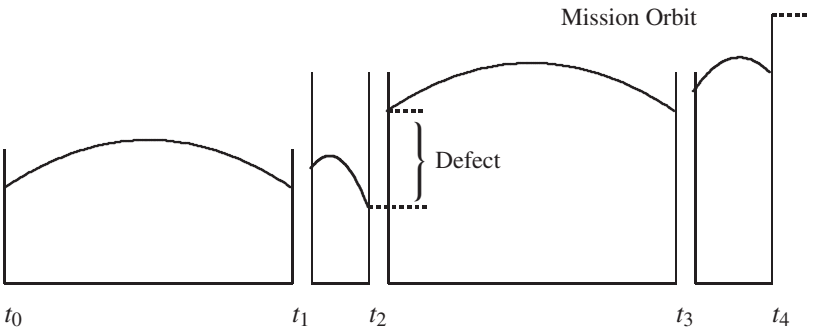
Table 6.7. *Minimum Δv transfer.*

	Formulation A	Formulation B
t_1	2690.41	*
α_1	179.667	179.667
Δv_1	8049.57	8049.57
θ_1	0.148637×10^{-2}	0.146647×10^{-2}
ψ_1	−9.08446	−9.08445
t_3	21658.5	*
α_2	180.063	180.063
Δv_2	5854.61	5854.61
θ_2	-0.136658×10^{-2}	-0.135560×10^{-2}
ψ_2	49.7892	49.7892
$\Delta v_1 + \Delta v_2$	13904.18221	13904.18220
FE	294	249

respectively). The only apparent difference between these approaches is that the formulation B requires fewer function evaluations (FE), which is not surprising because there are fewer variables. The equivalent weight delivered to the final orbit can be computed using (6.61) and is $w(t_F) = 0.23680470963252$ lb.

6.4.2 Multiple Shooting Formulation

Example 6.7 TWO-BURN MULTIPLE SHOOTING. A very natural partition of the problem is suggested by the physics of this two-burn orbit transfer. Modeling the dynamics using four distinct phases, namely the coast in the park orbit, the first burn, the coast in the transfer orbit, and the second burn, is an obvious choice. We can also treat each of these phases as segments and apply the *multiple shooting* method as described in Section 3.4. This is illustrated below:



For the multiple shooting formulation, one must augment the NLP variables (6.75) to include the state on each side of the segment (phase) boundaries as illustrated. Thus, we include as new NLP variables the states $\mathbf{r}(t_k^-), \mathbf{v}(t_k^-)$ and $\mathbf{r}(t_k^+), \mathbf{v}(t_k^+)$ for $k = 1,2,3,4$, where $-$ denotes the quantity before the boundary and $+$ denotes the quantity after the boundary. Additional "defect" constraints must be imposed to ensure continuity across the segment

boundaries and also to guarantee that the analytically propagated state is consistent with the (new) guessed values. Thus, when compared to the direct shooting method, the number of variables increases from 10 to 63. However, the number of constraints also increases from 7 to 60, so that the total number of degrees of freedom is unchanged.

One of the primary reasons to use a multiple shooting method is to improve robustness, which is demonstrated even on this simple four-segment example. As shown in Table 6.8, the multiple shooting method solved the problem with fewer gradient evaluations (2), fewer Hessian evaluations (3), and fewer function evaluations (97) when compared to the simple shooting approach. Even the modest increase in CPU time (attributed to increased problem size) is somewhat deceiving because the cost of evaluating the functions is so small for this example.

Table 6.8. *Shooting versus multiple shooting.*

	Shooting	Multiple shooting	Δ
GE	11	9	−22%
HE	5	2	−150%
FE	294	197	−49%
CPU (sec)	0.699127	0.872283	+20%

6.4.3 Collocation Formulation

Example 6.8 TWO-BURN COLLOCATION. Both Example 6.6 and Example 6.7 used approximate *physics*, that is, simplified orbit dynamics and impulsive burn approximations. For comparison, let us now solve the same orbit transfer without making these simplifying assumptions. We still pose the problem using four phases. However, in this example we consider finite-duration burn phases with optimal steering. Oblate gravitational accelerations will be used throughout, and we will describe the dynamics using the previously discussed equinoctial coordinates. During the coast phases, from (6.31), the dynamics are given by

$$\dot{\mathbf{y}} = \mathbf{A}(\mathbf{y})\boldsymbol{\Delta}_g + \mathbf{b}, \tag{6.77}$$

where $\mathbf{A}(\mathbf{y})$ is defined by (6.35), $\mathbf{b}$ by (6.36), and $\boldsymbol{\Delta}_g$ by (6.50). During the burn phases,

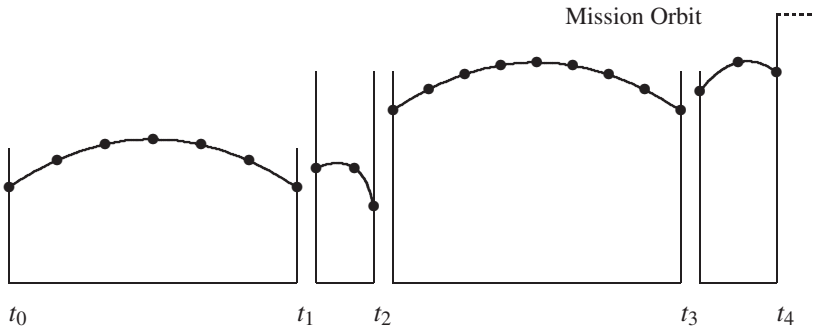
$$\dot{\mathbf{y}} = \mathbf{A}(\mathbf{y})\boldsymbol{\Delta} + \mathbf{b}, \tag{6.78}$$

$$\dot{w} = -T/I_{sp}, \tag{6.79}$$

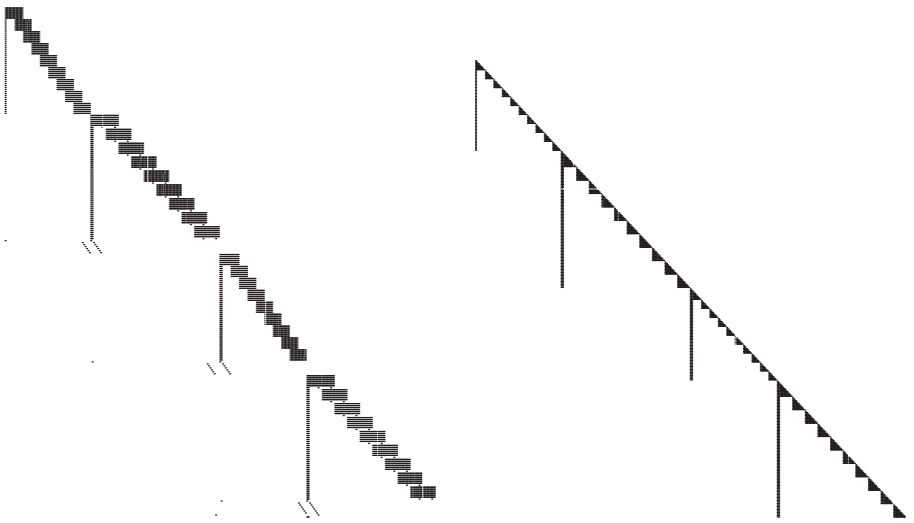
and the total perturbation is given by (6.44). To be consistent with Examples 6.6 and 6.7, we define the orientation of the thrust using pitch and yaw angles in the inertial velocity frame and then transform them to the radial frame, that is,

$$\boldsymbol{\Delta}_T = \mathbf{Q}_r^{\mathsf{T}}\mathbf{Q}_v \begin{bmatrix} T\cos\theta\cos\psi \\ T\cos\theta\sin\psi \\ T\sin\theta \end{bmatrix}, \tag{6.80}$$

where $\mathbf{Q}_v$ is defined by (6.63) and $\mathbf{Q}_r$ is given by (6.45). Since the dynamics are represented using equinoctial coordinates, it is convenient to also specify the boundary conditions in these coordinates. Thus, the geosynchronous conditions (6.70)–(6.74) are equivalent to having $p = 19323/\sigma + R_e$ and $f = g = h = k = 0$. Constraints are also used to link the equinoctial states across the phase boundaries as with the multiple shooting formulation. The weight at the end of phase 2 is also constrained to equal the weight at the beginning of phase 4. The situation is illustrated below:



SOCS was used to solve the problem beginning with a trapezoidal discretization and 10 grid points per phase. The first NLP has 307 variables and 268 active constraints, or 39 degrees of freedom. Figure 6.20 displays the sparsity pattern of the Jacobian and Hessian matrices, corresponding to the initial trapezoidal discretization with 10 grid points per phase. Note that, because the Hessian is symmetric, only the lower-triangular portion is



(a) Jacobian (b) Hessian

Figure 6.20. *Sparsity patterns.*

shown. The iteration history is summarized in Table 6.9. The second and third columns in the table give a phase-by-phase breakdown of the discretization method (Disc.) and number of grid points (M), respectively. Trapezoidal discretization is denoted by T and separated Simpson by H. Observe that the mesh-refinement algorithm tends to concentrate a large number of points in the early portions of the trajectory because the oblate earth perturbations are more significant in this region. The final mesh-refinement iteration required solving a sparse NLP problem with 5149 variables, 4714 active constraints, and 435 degrees of freedom.

Table 6.9. *Two-burn orbit transfer performance summary.*

Iter.	Disc.	M	FE	$\epsilon_{\max}$	T (sec)
1	(T,T,T,T)	(10,10,10,10)	2164	0.26×10^{0}	0.41×10^{2}
2	(H,T,T,T)	(10,19,16,19)	604	0.53×10^{-2}	0.20×10^{2}
3	(H,H,H,H)	(19,19,16,19)	526	0.14×10^{-2}	0.32×10^{2}
4	(H,H,H,H)	(37,37,31,37)	137	0.96×10^{-5}	0.24×10^{2}
5	(H,H,H,H)	(73,73,61,37)	113	0.36×10^{-6}	0.35×10^{2}
6	(H,H,H,H)	(145,73,121,37)	113	0.59×10^{-7}	0.49×10^{2}
Total	-	376	3657		201.84

Figure 6.21 illustrates the optimal two-burn transfer to geosynchronous orbit. The optimal steering angles for both burns are plotted in Figure 6.22 with the times normalized to the beginning of the burn. It is interesting to observe that the optimal burn times, i.e., the lengths of phases 2 and 4, are $t_2 - t_1 = 141.47$ (sec) and $t_4 - t_3 = 49.40$ (sec). When compared to the total mission time of 21683.5 (sec), these burn times are quite short, hence justifying the impulsive approximation. It is also interesting to note that the "true" optimal objective $w(t_F) = 0.2367248713$ lb is only 0.033% less than the impulsive burn analytic coast approximation computed in Example 6.6. However, it is necessary to use the optimal steering in Figure 6.22 to achieve this performance.

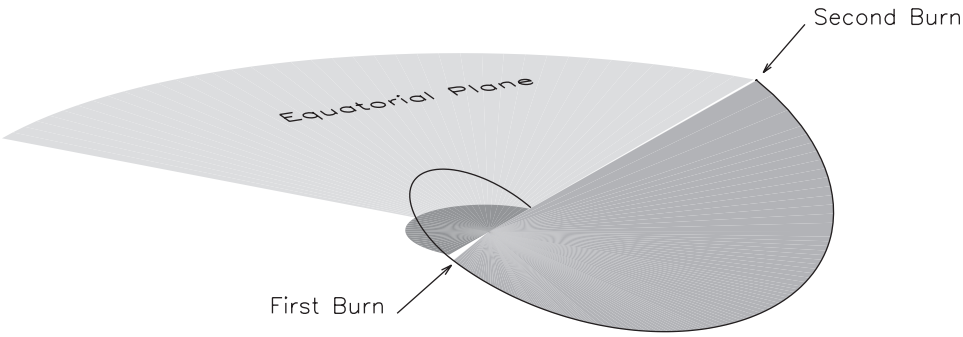
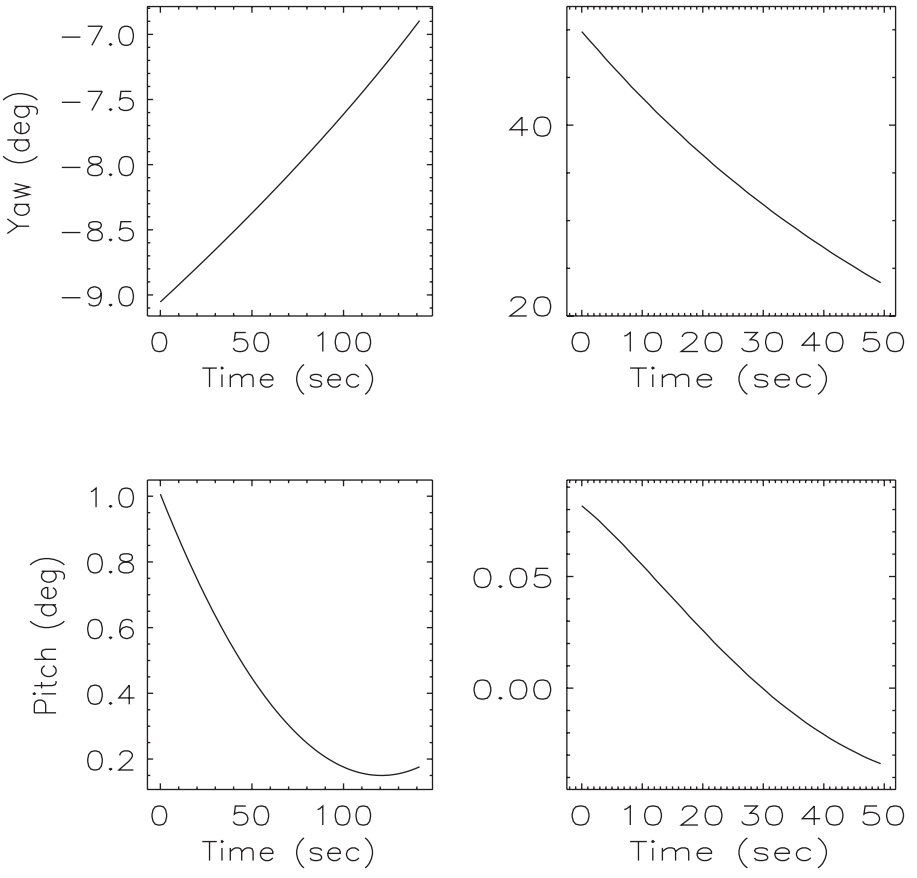


Figure 6.21. *Two-burn orbit transfer.*



(a) First burn (b) Second burn

Figure 6.22. *Optimal control angles.*

6.5 Hang Glider

Example 6.9 RANGE MAXIMIZATION OF A HANG GLIDER. Originally posed by Bulirsch et al. [58], this problem describes the optimal control of a hang glider in the presence of a specified thermal updraft. It is particularly sensitive to the accuracy of the mesh, a difficulty which was resolved in the reference by exploiting a combination of direct and indirect methods.

The state variables are $\mathbf{y}^{\mathsf{T}}(t) = (x, y, v_x, v_y)$, where x is the horizontal distance (meters), y the altitude (meters), v_x the horizontal velocity (meters/sec), and v_y the vertical velocity (meters/sec). The control variable is $u(t) = C_L$, the aerodynamic lift coefficient. The final time t_F is free and the final range x_F is to be maximized. The state equations

which describe the planar motion for the hang glider are

$$\dot{x} = v_x, \tag{6.81}$$
$$\dot{y} = v_y, \tag{6.82}$$
$$\dot{v_x} = \frac{1}{m}(-L\sin\eta - D\cos\eta), \tag{6.83}$$
$$\dot{v_y} = \frac{1}{m}(L\cos\eta - D\sin\eta - W), \tag{6.84}$$

where the quadratic drag polar

$$C_D(C_L) = C_0 + kC_L^2 \tag{6.85}$$

with

$$D = \frac{1}{2}C_D\rho S v_r^2, \qquad L = \frac{1}{2}C_L\rho S v_r^2,$$
$$X = \left(\frac{x}{R} - 2.5\right)^2, \qquad u_a(x) = u_M(1-X)\exp[-X],$$
$$V_y = v_y - u_a(x), \qquad v_r = \sqrt{v_x^2 + V_y^2},$$
$$\sin\eta = \frac{V_y}{v_r}, \qquad \cos\eta = \frac{v_x}{v_r}.$$

Note the results presented in [58] correspond to a value of 3.5 instead of 2.5 in the expression for X. The following constants complete the definition of the problem:

$$u_M = 2.5, \qquad m = 100\,(\text{kg}),$$
$$R = 100, \qquad S = 14\,(\text{m}^2),$$
$$C_0 = .034, \qquad \rho = 1.13\,(\text{kg/m}^3),$$
$$k = .069662, \qquad g = 9.80665\,(\text{m/sec}^2),$$

where $W = mg$. The lift coefficient is bounded

$$0 \le C_L \le 1.4 \tag{6.86}$$

and the following boundary conditions are imposed:

$$x(0) = 0\,(\text{m}), \qquad x(t_F): \quad \text{free},$$
$$y(0) = 1000\,(\text{m}), \qquad y(t_F) = 900\,(\text{m}),$$
$$v_x(0) = 13.227567500\,(\text{m/sec}), \qquad v_x(t_F) = 13.227567500\,(\text{m/sec}),$$
$$v_y(0) = -1.2875005200\,(\text{m/sec}), \qquad v_y(t_F) = -1.2875005200\,(\text{m/sec}).$$

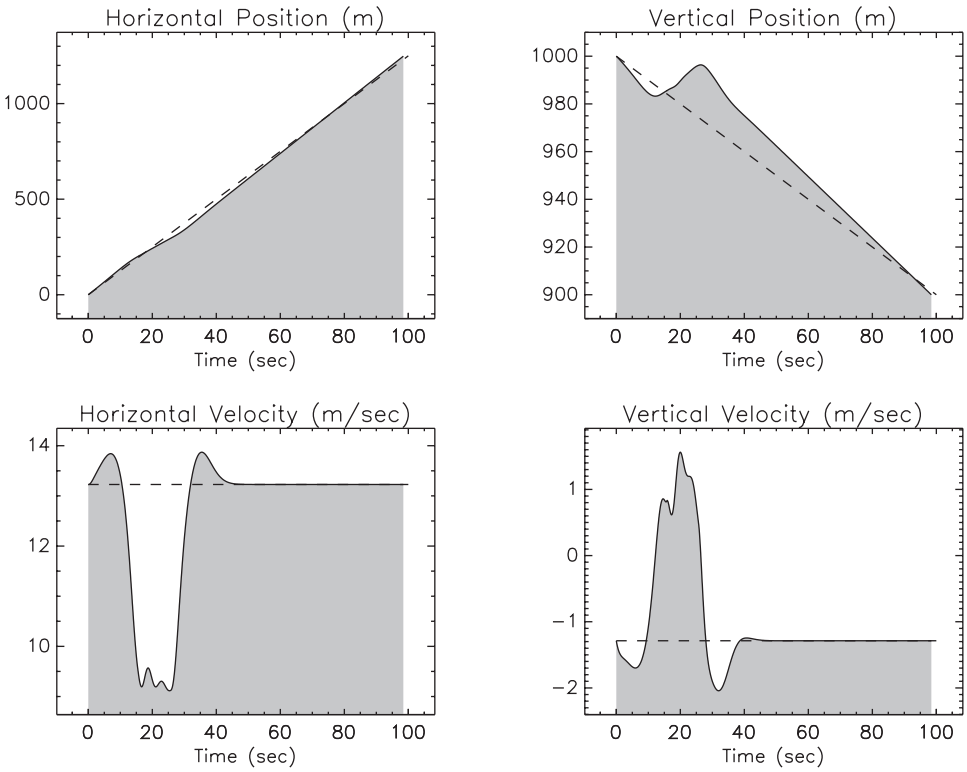


Figure 6.23. *Hang glider, states.*

The initial guess was computed using linear interpolation between the boundary conditions, with $x(t_F) = 1250$, and $C_L(0) = C_L(t_F) = 1$.

The optimal state histories are displayed in Figure 6.23, with the initial guess plotted as a dashed line. Figure 6.24 shows the optimal control. The mesh-refinement history is summarized in Table 6.10 and plotted in Figure 6.25. Observe that eight mesh-refinement iterations were required before the discretization error ϵ was reduced significantly. This behavior is consistent with the sensitivity reported in [58], which led them to use a hybrid technique. The maximum range is $x^*(t_F) = 1248.031026$ (m).

6.6 Abort Landing in the Presence of Windshear

Example 6.10 ABORT LANDING IN THE PRESENCE OF WINDSHEAR. The dynamic behavior of an aircraft landing in the presence of a windshear was first formulated as an optimal control problem by Miele,[4] Wang, and Melvin [134]. A number of other authors

[4]Dr. Angelo Miele was formerly Director of Astrodynamics and Flight Mechanics at Boeing Scientific Research Laboratory (BSRL), and is now Foyt Professor Emeritus at Rice University.

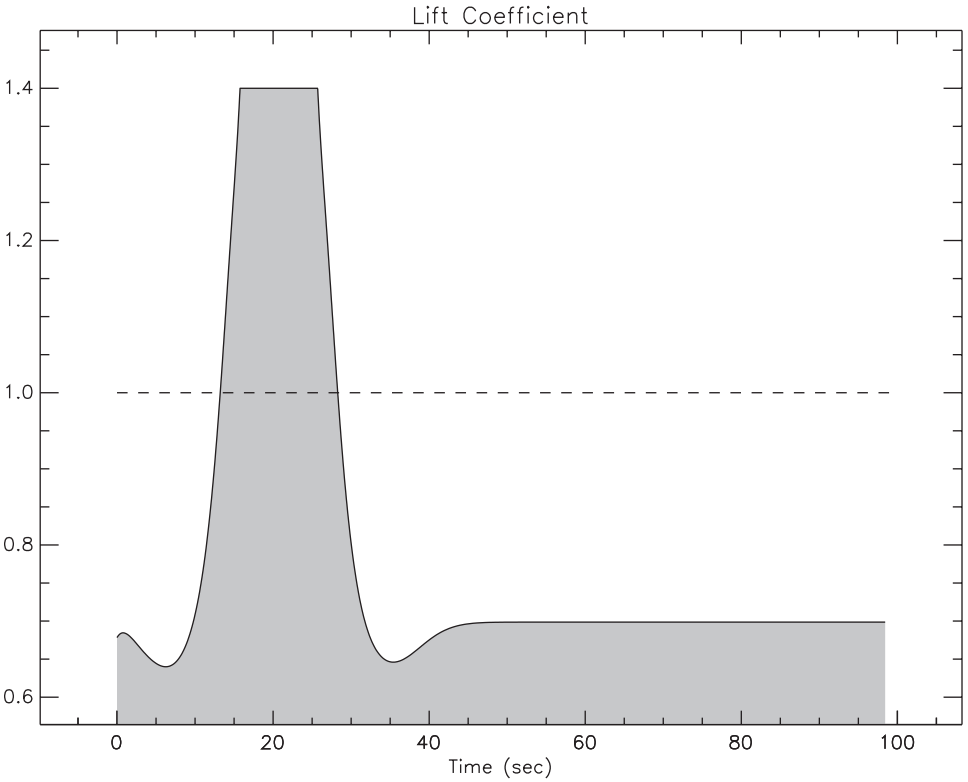


Figure 6.24. *Hang glider control $C_L(t)$.*

Table 6.10. *Hang glider mesh-refinement summary.*

k	M	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	25	23	16	383	14207	2.0200	1.00×10^{-1}
2	49	30	28	605	38163	3.5669×10^{1}	2.90×10^{-1}
3	49	31	29	1946	204480	1.2682×10^{1}	7.00×10^{-1}
4	97	56	54	3596	722034	5.2137	2.13
5	106	18	16	1088	259858	5.5682	8.30×10^{-1}
6	113	33	31	2078	499588	1.4190	1.67
7	134	22	20	1352	398490	1.2231	1.30
8	145	17	15	1022	335652	1.2494	1.15
9	151	17	15	1022	349236	1.7212×10^{-2}	1.20
10	277	21	19	1286	777980	1.6366×10^{-2}	3.22
11	291	9	6	449	330507	3.4776×10^{-5}	1.23
12	567	4	2	164	312058	5.2638×10^{-7}	3.02
13	739	3	1	98	306696	2.0964×10^{-8}	4.04
Total	739	284	252	15089	4548949		$2.088\times10^{+1}$

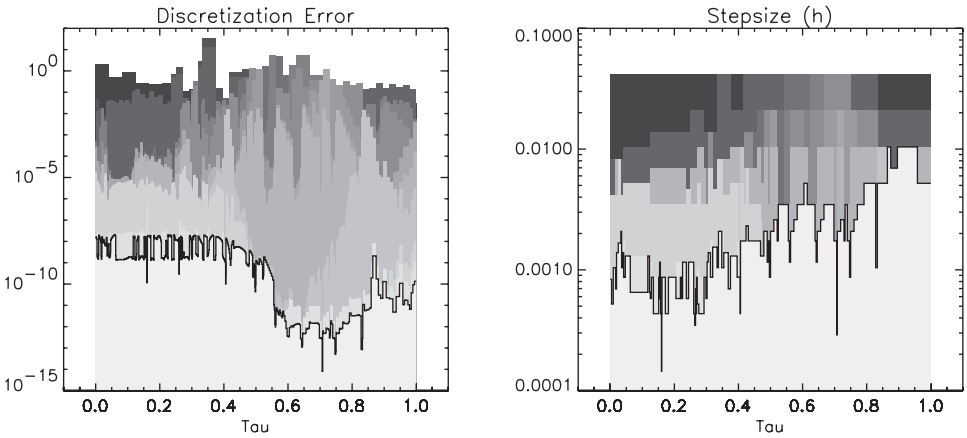


Figure 6.25. *Hang glider mesh-refinement history.*

investigated the problem including Bulirsch, Montrone, and Pesch [56, 57], who introduce the problem as follows:

> One of the most dangerous situations for a passenger aircraft in take-off and landing is caused by the presence of low altitude windshears. This meteorological phenomenon, which is more common in subtropical regions, is usually associated with high ground temperatures leading to a so-called downburst. This downburst involves a column of descending air which spreads horizontally near the ground. Even for a highly skilled pilot, an inadvertent encounter with a windshear can be a fatal problem, since the aircraft might encounter a headwind followed by a tailwind, both coupled with a downdraft. The transition from headwind to tailwind yields an acceleration so that the resulting windshear inertia force can be as large as the drag of the aircraft, and sometimes as large as the thrust of the engines. This explains why the presence of low altitude windshears is a threat to safety in aviation. Some 30 aircraft accidents over the past 20 years have been attributed to windshear, and this attests to the perilousness of this occurrence. Among these accidents, the most disastrous ones happened in 1982 in New Orleans, where 153 people were killed, and in 1985 in Dallas, where 137 people were killed.

6.6.1 Dynamic Equations

Following the development in [56] the dynamic behavior of the aircraft is described by

$$\dot{x} = v\cos\gamma + w_x, \tag{6.87}$$

$$\dot{h} = v\sin\gamma + w_h, \tag{6.88}$$

$$\dot{v} = \frac{1}{m}[T\cos(\alpha+\delta) - D] - g\sin\gamma - (\dot{w}_x\cos\gamma + \dot{w}_h\sin\gamma), \tag{6.89}$$

$$\dot{\gamma} = \frac{1}{mv}[T\sin(\alpha+\delta) + L] - \frac{g}{v}\cos\gamma + (\dot{w}_x\sin\gamma - \dot{w}_h\cos\gamma), \tag{6.90}$$

where the state variables are the horizontal distance x, the altitude h, the relative velocity v, and the relative flight path angle γ. The thrust and aerodynamic forces are defined by

$$T = \beta T_*, \tag{6.91}$$

$$T_* = a_0 + a_1 v + a_2 v^2, \tag{6.92}$$

$$D = \frac{1}{2} C_D \rho S v^2, \tag{6.93}$$

$$C_D(\alpha) = b_0 + b_1 \alpha + b_2 \alpha^2, \tag{6.94}$$

$$L = \frac{1}{2} C_L \rho S v^2, \tag{6.95}$$

$$C_L(\alpha) = \begin{cases} c_0 + c_1 \alpha, & \alpha \le \alpha_*, \\ c_0 + c_1 \alpha + c_2 (\alpha - \alpha_*)^2, & \alpha_* \le \alpha \le \alpha_{max}, \end{cases} \tag{6.96}$$

$$\beta(t) = \begin{cases} \beta_0 + \dot{\beta}_0 t, & 0 \le t \le t_\beta, \\ 1, & t_\beta \le t \le t_F, \end{cases} \tag{6.97}$$

where the thrust, drag, and lift are denoted by T, D, and L, respectively. The power setting β is specified as in [134]. It is hypothesized that upon sensing a downdraft, the pilot increases power at a constant rate until reaching a maximum value at time $t_\beta = (1 - \beta_0)/\dot{\beta}_0$ and thereafter holds it constant. The windshear is modeled as follows:

$$w_x = A(x), \tag{6.98}$$

$$w_h = \frac{h}{h_*} B(x) \tag{6.99}$$

with

$$A(x) = \begin{cases} -50 + ax^3 + bx^4, & 0 \le x \le 500, \\ \frac{1}{40}(x - 2300), & 500 \le x \le 4100, \\ 50 - a(4600 - x)^3 - b(4600 - x)^4, & 4100 \le x \le 4600, \\ 50, & 4600 \le x, \end{cases} \tag{6.100}$$

$$B(x) = \begin{cases} dx^3 + ex^4, & 0 \le x \le 500, \\ -51 \exp[-c(x - 2300)^4], & 500 \le x \le 4100, \\ d(4600 - x)^3 - e(4600 - x)^4, & 4100 \le x \le 4600, \\ 0, & 4600 \le x. \end{cases} \tag{6.101}$$

Table 6.11 summarizes the various parameters as given in [56] needed to complete the model of a Boeing 727 airplane.

Table 6.11. *Dynamic model parameters.*

t_F	40 sec	u_{max}	3 deg/sec
α_{max}	17.2 deg	ρ	$.2203\times10^{-2}$ lb sec^2 ft^{-4}
S	$.1560\times10^{4}$ ft^2	g	3.2172×10^{1} ft sec^{-2}
mg	150000 lb	δ	2 deg
a_0	$.4456\times10^{5}$ lb	a_1	$-.2398\times10^{2}$ lb sec/ft
a_2	$.1442\times10^{-1}$ lb sec^2 ft^{-2}	β_0	.3825
$\dot{\beta}_0$	.2 sec^{-1}	b_0	.1552
b_1	.12369 rad^{-1}	b_2	2.4203 rad^{-2}
c_0	.7125	c_1	6.0877 rad^{-1}
c_2	-9.0277 rad^{-2}	a_*	12 deg
h_*	1000 ft	a	6×10^{-8} sec^{-1} ft^{-2}
b	-4×10^{-11} sec^{-1} ft^{-3}	c	$-\ln(25/30.6)\times10^{-12}$ ft^{-4}
d	-8.02881×10^{-8} sec^{-1} ft^{-2}	e	6.28083×10^{-11} sec^{-1} ft^{-3}
x_0	0 ft	γ_0	-2.249 deg
h_0	600 ft	α_0	7.353 deg
v_0	239.7 ft/sec	γ_F	7.431 deg

6.6.2 Objective Function

The goal of this abort landing problem is to avoid having the airplane crash. To achieve this we introduce two things: an optimization parameter h_{min}, which is the minimum altitude that occurs during the landing, and a new path inequality constraint

$$h(t) \geq h_{min}. \tag{6.102}$$

The optimization objective is to *maximize*

$$F = h_{min} \tag{6.103}$$

subject to the path inequality (6.102). This *maximin* problem guarantees the aircraft will be as high above the ground as possible. The results obtained using $\mathbb{SOCS}$ treat this formulation directly.

In contrast, both [134] and [56] treat the objective differently. Because of the relationship between the Hölder and Chebyshev norm

$$\lim_{k\to\infty}\left[\int_0^{t_F}[h_r - h(t)]^{2k}dt\right]^{\frac{1}{2k}} = \max_{0\leq t\leq t_f}[h_r - h(t)],$$

where $h_r = 1000$ ft is a reference altitude. Specifically, Miele, Wang, and Melvin [134] simply choose a large value for k and minimize

$$F_M = \int_0^{t_F}[h_r - h(t)]^6 dt.$$

Unfortunately this approach is numerically unattractive because of the scale of the objective function. In order to address the numeric difficulties Bulirsch, Montrone, and Pesch [56]

introduce a new state variable and path inequality constraint

$$\dot{\zeta} = 0,$$
$$0 \geq h_r - h(t) - \zeta(t)$$

and then postulate a composite objective function

$$F_B = \Lambda \int_0^{t_F} [h_r - h(t)]^6 dt + \Theta\zeta(t_F).$$

The parameter Λ is gradually changed from one to zero, while Θ is increased from zero to one using a homotopy strategy described in [57].

6.6.3 Control Variable

The control variable for this problem is the angle of attack $\alpha(t)$, which is subject to the constraints

$$\alpha(t) \leq \alpha_{max}, \tag{6.104}$$
$$|\dot{\alpha}(t)| \leq u_{max}. \tag{6.105}$$

Using the approach described in Section 4.10 it is possible to directly impose the rate constraint (6.105).

In contrast, the angle of attack is treated as a state variable in [56] and the rate is treated as a control. This introduces an additional differential equation

$$\dot{\alpha} = u$$

in which case the rate constraint (6.105) can be treated as simple bounds $-u_{max} \leq u \leq u_{max}$ on the (new) control. Unfortunately, this transformation introduces two other more deleterious effects. The new control u appears linearly in the problem, whereas the "real" control α appears nonlinearly, and consequently singular arcs can (and do) occur. Furthermore, it increases the order of the path constraints. When u is the control, the solution has a boundary arc with respect to the first order state constraint and touch points with respect to the third order state constraints.

6.6.4 Model Data

The mathematical description of a complicated physical system usually entails specification of tabular data. In order to construct a mathematical model with the appropriate continuity, one approach is to construct a B-spline approximation as described in Section 6.2.3. This becomes particularly important when using an indirect method to solve the optimal control problem as in [56] because discontinuities introduce "interior-point conditions" and as they state

> ...a reduction of the matching points from 8 points in Miele's model to 3 points in our model, makes the derivation of the necessary conditions of optimal control theory much easier. Since Miele's model uses cubic splines, the wind components of his model are C^2 functions everywhere.

While smoothing the model data is one way to treat discontinuous functions, there is another approach which can be employed. Let us model the problem using a multiple phase structure. Since discontinuous behavior can occur at a phase boundary, we can define a phase structure specifically to treat the different regions in the problem data. In particular let us model the problem using five distinct phases. Each phase is characterized by a unique set of boundary conditions and right-hand-side functions as follows:

Phase 1 $\boxed{0 = t_I^{(1)} \le t \le t_F^{(1)}}$

$$x[t_I^{(1)}] = x_0, \qquad h[t_I^{(1)}] = h_0,$$
$$v[t_I^{(1)}] = v_0, \qquad \gamma[t_I^{(1)}] = \gamma_0,$$
$$\alpha[t_I^{(1)}] = \alpha_0, \qquad x[t_F^{(1)}] = 500,$$
$$A(x) = -50 + ax^3 + bx^4, \qquad B(x) = dx^3 + ex^4,$$
$$\beta(t) = \beta_0 + \dot{\beta}_0 t.$$

Phase 2 $\boxed{t_I^{(2)} \le t \le t_F^{(2)} = t_\beta}$

$$x[t_I^{(2)}] = 500, \qquad \beta(t) = \beta_0 + \dot{\beta}_0 t,$$
$$A(x) = -50 + ax^3 + bx^4, \qquad B(x) = dx^3 + ex^4.$$

Phase 3 $\boxed{t_\beta = t_I^{(3)} \le t \le t_F^{(3)}}$

$$x[t_F^{(3)}] = 4100, \qquad \beta(t) = 1,$$
$$A(x) = \frac{1}{40}(x - 2300), \qquad B(x) = -51\exp[-c(x-2300)^4].$$

Phase 4 $\boxed{t_I^{(4)} \le t \le t_F^{(4)}}$

$$x[t_I^{(4)}] = 4100, \qquad x[t_F^{(4)}] = 4600,$$
$$A(x) = 50 - a(4600 - x)^3 - b(4600 - x)^4, \qquad B(x) = d(4600 - x)^3 - e(4600 - x)^4,$$
$$\beta(t) = 1.$$

Phase 5 $\boxed{t_I^{(5)} \le t \le t_F^{(5)} = t_F}$

$$x[t_I^{(5)}] = 4600, \qquad \gamma(t_F^{(5)}) = \gamma_F,$$
$$A(x) = 50, \qquad B(x) = 0,$$
$$\beta(t) = 1.$$

Observe that the throttle parameter (6.97) is a monotonic function of time, and the wind model (6.100)–(6.101) is a monotonic function of distance x which in turn is a monotonic function of time. Consequently there is a natural "interleaving" of the discontinuous regions that is explicitly identified by the phase structure. Note also that the discontinuous behavior in the second derivative of the lift coefficient (6.96) does not present any difficulties because it is a function of the control α, which can be discontinuous.

6.6.5 Computational Results

The overall problem can be posed using five distinct phases. On all five phases, there are four state variables $\mathbf{y}^{\mathsf{T}} = (x, h, v, \gamma)$, one control $u = \alpha$, and one parameter $p = h_{min}$. Each phase must satisfy the differential equations (6.87)–(6.90) and the path constraint (6.102). On all phases, the control variable must satisfy the simple bounds (6.104) and the rate constraints (6.105). At all interior phase boundaries we require continuity in the state, control, parameter, and time, unless the values are fixed by the boundary conditions. Within each individual phase, the right-hand-side functions are defined by the explicit phase structure. And finally the goal is to maximize the parameter $p = h_{min}$ (in the last phase).

Using a linear guess for the dynamic variables, with 10 grid points in each phase, the optimal control problem was solved using $\mathbb{SOCS}$ and the computational performance is summarized in Table 6.12. The requested resolution for the mesh-refinement procedure was $\epsilon_{\max} \leq 10^{-7}$, which was obtained after seven mesh-refinement iterations. The trapezoidal (TR) discretization was used for the early iterations, followed by either the compressed Hermite–Simpson (HC) or the separated Hermite–Simpson (HS) method, as indicated in the second column of the table. The number of grid points is given in the third column, the number of QP subproblems in the fourth column, and the accuracy achieved in the fifth column. The total CPU time for each iteration is given in the last column, with the entire solution obtained in 9.00 sec.

Table 6.12. *Windshear performance summary.*

Iter.	Disc.	M	NQP	$\epsilon_{\max}$	CPU (sec)
1	(TR,TR,TR,TR,TR)	(10,10,10,10,10)	10	2.29×10^{-2}	8.0×10^{-2}
2	(TR,HC,TR,HC,TR)	(19,10,19,10,19)	11	2.18×10^{-3}	1.6×10^{-1}
3	(HS,HC,HS,HC,HS)	(37,10,37,19,37)	9	3.02×10^{-4}	3.1×10^{-1}
4	(HS,HS,HS,HS,HS)	(67,19,73,47,73)	7	2.69×10^{-5}	6.6×10^{-1}
5	(HS,HS,HS,HS,HS)	(67,19,115,47,145)	19	2.67×10^{-6}	4.8×10^{0}
6	(HS,HS,HS,HS,HS)	(67,19,229,47,145)	7	1.32×10^{-7}	1.9×10^{0}
7	(HC,HC,HC,HC,HC)	(34,10,120,24,73)	4	9.87×10^{-8}	1.0×10^{0}
Total	-	261	67		9.00

Figure 6.26 illustrates the optimal trajectory, and the optimal time histories for the four state variables are plotted in Figure 6.27. The phase boundaries are delineated by dotted vertical lines, and the lower bound on the altitude is shown with a dashed horizontal line. The optimal value achieved was $F^* = 491.85230$ (ft). The illustration suggests that there is a region in Phase 3 during which the altitude is on the minimum altitude boundary and a second touch point in Phase 5. In fact, during the extended boundary arc in Phase 3, the state inequality is *not* in the active set at all grid points. Instead there is a very

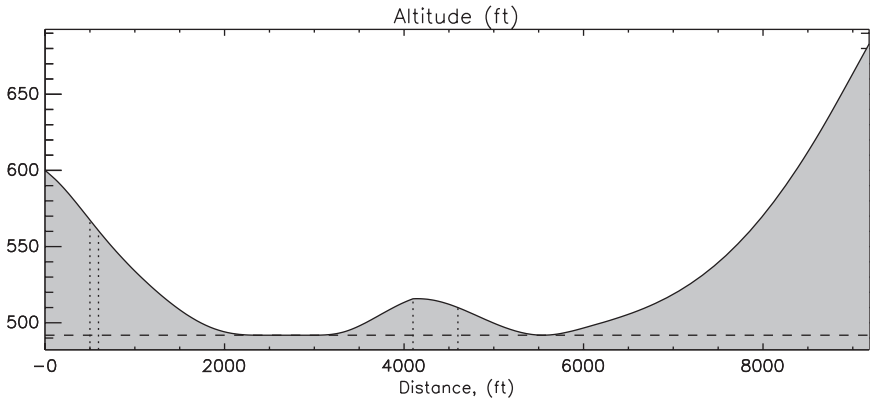


Figure 6.26. *Optimal trajectory profile.*

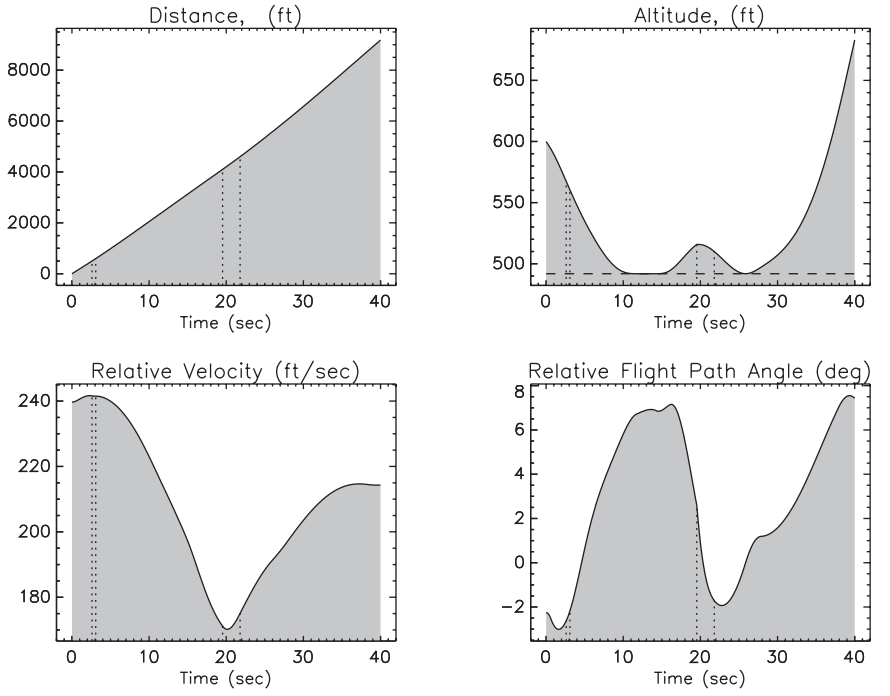


Figure 6.27. *Optimal state profile.*

small oscillation in the altitude, all within the mesh-refinement tolerance. This behavior is similar to that discussed in Section 4.12 (cf. Figure 4.29). Figure 6.28 illustrates the optimal control required to maximize the objective, as well as the corresponding angle of attack rate.

It is worthwhile to compare the results presented here with those found in [56]. Qualitatively the results are very similar, with the optimal trajectory having an arc that follows

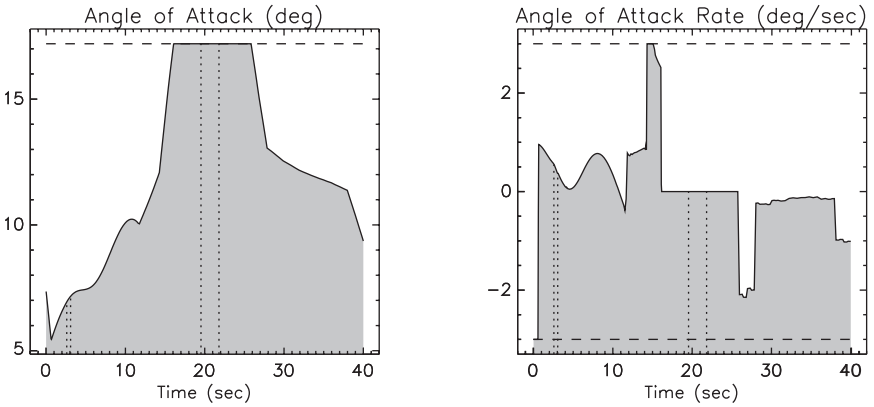


Figure 6.28. *Optimal control profile.*

the minimum altitude, and a subsequent single touch point. The angle of attack profile also compares favorably. Nevertheless, the solution techniques differ in three ways:

1. treatment of the objective function,

2. definition of the control variable,

3. treatment of the physical model data.

In particular, in [56], two additional states are introduced: one because $\dot{\alpha}$ is treated as the control, and a second to represent the minimum altitude. An indirect multiple shooting method was used to solve this problem, which required a sophisticated homotopy strategy in order to identify the correct switching structure and also deal with singular arcs. In contrast, the approach described here does not introduce additional state variables, and as such does not require the same derivative continuity in the angle of attack. Finally, our approach does not increase the index of the DAE, i.e., the order of the state constraints, and avoids the computational complexity associated with picking the correct arc structure as required by an indirect method.

6.7 Space Station Attitude Control

Example 6.11 SPACE STATION ATTITUDE CONTROL. In his Master's thesis on the pseudospectral method, Pietz [142] presents results for an application that arises when trying to control the attitude of the International Space Station. A complete discussion of the problem can be found in Chapter 4 of the thesis. For convenience we present a brief summary of the optimal control problem. The state of the system is defined by three vectors, namely $\boldsymbol{\omega}(t)$, which is the angular velocity of the spacecraft with respect to an inertial reference frame, $\mathbf{r}(t)$, which are the Euler–Rodriguez parameters used to define the vehicle attitude, and $\mathbf{h}(t)$, which is the angular momentum of the control moment gyroscopes (CMGs). The torque $\mathbf{u}(t)$ is used to control the system. The dynamics are given by the

differential-algebraic system

$$\dot{\boldsymbol{\omega}} = \mathbf{J}^{-1}\left\{\boldsymbol{\tau}_{gg}(\mathbf{r}) - \boldsymbol{\omega}(t)^{\otimes}\left[\mathbf{J}\boldsymbol{\omega}(t) + \mathbf{h}(t)\right] - \mathbf{u}(t)\right\}, \tag{6.106}$$

$$\dot{\mathbf{r}} = \frac{1}{2}\left[\mathbf{r}(t)\mathbf{r}^{\mathsf{T}}(t) + \mathbf{I} + \mathbf{r}(t)^{\otimes}\right]\left[\boldsymbol{\omega}(t) - \boldsymbol{\omega}_0(\mathbf{r})\right], \tag{6.107}$$

$$\dot{\mathbf{h}} = \mathbf{u}(t), \tag{6.108}$$

$$0 \le h_{max} - \|\mathbf{h}\|. \tag{6.109}$$

The dynamics utilize the skew-symmetric cross product operator defined by

$$\mathbf{a}^{\otimes} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{6.110}$$

The gravity gradient torque is given by

$$\boldsymbol{\tau}_{gg}(\mathbf{r}) = 3\omega_{orb}^2 \mathbf{C}_3^{\otimes}\mathbf{J}\mathbf{C}_3, \tag{6.111}$$

where $\mathbf{C}_3$ is the third column of the rotation matrix

$$\mathbf{C} = \mathbf{I} + \frac{2}{1 + \mathbf{r}^{\mathsf{T}}\mathbf{r}}\left(\mathbf{r}^{\otimes}\mathbf{r}^{\otimes} - \mathbf{r}^{\otimes}\right). \tag{6.112}$$

Also

$$\boldsymbol{\omega}_0(\mathbf{r}) = -\omega_{orb}\mathbf{C}_2. \tag{6.113}$$

The state at the initial time is prescribed leading to the boundary conditions

$$\boldsymbol{\omega}(t_0) = \overline{\boldsymbol{\omega}}_0, \tag{6.114}$$

$$\mathbf{r}(t_0) = \overline{\mathbf{r}}_0, \tag{6.115}$$

$$\mathbf{h}(t_0) = \overline{\mathbf{h}}_0. \tag{6.116}$$

At the final time it is desirable to reach an orientation that can be maintained without utilizing additional control torque. This torque equilibrium attitude (TEA) is achieved by setting the right-hand sides of (6.106) and (6.107) to zero, with $\mathbf{u} = \mathbf{0}$, leading to the boundary conditions

$$\mathbf{0} = \mathbf{J}^{-1}\left\{\boldsymbol{\tau}_{gg}(\mathbf{r}(t_f)) - \boldsymbol{\omega}(t_f)^{\otimes}\left[\mathbf{J}\boldsymbol{\omega}(t_f) + \mathbf{h}(t_f)\right]\right\}, \tag{6.117}$$

$$\mathbf{0} = \frac{1}{2}\left[\mathbf{r}(t_f)\mathbf{r}^{\mathsf{T}}(t_f) + \mathbf{I} + \mathbf{r}(t_f)^{\otimes}\right]\left[\boldsymbol{\omega}(t_f) - \boldsymbol{\omega}_0(\mathbf{r}(t_f))\right]. \tag{6.118}$$

The goal is to choose the control vector $\mathbf{u}(t)$ to minimize the magnitude of the final momentum $\|\mathbf{h}\|$, or equivalently to minimize

$$F = \mathbf{h}^{\mathsf{T}}(t_f)\mathbf{h}(t_f). \tag{6.119}$$

Note that Pietz [142] treats $\|\mathbf{h}(t_f)\|$ as the objective function; however, this quantity is not differentiable at $\|\mathbf{h}(t_f)\| = 0$. In contrast, the objective given by (6.119) is differentiable everywhere and leads to an equivalent solution.

The problem description is completed by specifying the following parameters:

$$t_0 = 0, \qquad t_f = 1800,$$
$$h_{max} = 10000, \qquad \omega_{orb} = .06511\frac{\pi}{180},$$

$\overline{\boldsymbol{\omega}}_0$	$\overline{\mathbf{r}}_0$	$\overline{\mathbf{h}}_0$
$-9.5380685844896\times10^{-6}$	$2.9963689649816\times10^{-3}$	5000
$-1.1363312657036\times10^{-3}$	$1.5334477761054\times10^{-1}$	5000
$5.3472801108427\times10^{-6}$	$3.8359805613992\times10^{-3}$	5000

$$\mathbf{J} = \begin{pmatrix} 2.80701911616\times10^{7} & 4.822509936\times10^{5} & -1.71675094448\times10^{7} \\ 4.822509936\times10^{5} & 9.5144639344\times10^{7} & 6.02604448\times10^{4} \\ -1.71675094448\times10^{7} & 6.02604448\times10^{4} & 7.6594401336\times10^{7} \end{pmatrix}.$$

The optimal control problem defined by (6.106)–(6.119) was solved using the $\mathbb{SOCS}$ software. Mesh refinement was performed to ensure the discretization error was below 10^{-7}, which produces a solution with at least eight significant figures of accuracy. Constant values were used as initial guesses for all of the dynamic variables in the problem.

Unfortunately the original formulation suggested by Pietz [142] does not have a *unique* solution. In particular there are many control histories that will produce $F^* = \mathbf{h}^{*\mathsf{T}}(t_f)\mathbf{h}^*(t_f) = 0$. From an algorithmic perspective, when a nonunique problem is solved the standard recourse for the underlying nonlinear program is to *force* the projected Hessian matrix to be positive definite. Within $\mathbb{SOCS}$ this is achieved by introducing a nonzero Levenberg parameter to form a modified Hessian matrix. For the SNOPT algorithm used to obtain the results in Pietz [142], a quasi-Newton Hessian approximation is used which is forced to be positive definite. Thus in all cases the computational algorithms produce a solution; however, none of them is unique!

With this insight, the problem can be reformulated. One possible approach is to enforce the additional boundary conditions

$$\mathbf{h}(t_f) = \mathbf{0} \tag{6.120}$$

and then introduce a different objective that uniquely defines the optimal trajectory, e.g., minimize

$$F = 10^{-6}\int_{t_0}^{t_f} \mathbf{u}^{\mathsf{T}}(t)\mathbf{u}(t)dt. \tag{6.121}$$

This can be interpreted as a "minimum energy" criterion. Using this modified formulation, the problem was solved using $\mathbb{SOCS}$ and the solution is illustrated with a shaded region and solid line in Figures 6.29–6.31. The optimal value of the objective function is $F^* = 3.586883988$. It is also interesting that the computational behavior of the $\mathbb{SOCS}$ algorithm for this problem was quite well behaved as summarized in Table 6.13.

The thesis of Bhatt [41] extends the dynamic model proposed by Pietz to incorporate multibody effects and aerodynamic torques. Bhatt also suggests minimizing the peak CMG momentum as an objective function; i.e., instead of using (6.121) let us minimize

$$F = h_{max}, \tag{6.122}$$

which is the upper bound appearing in (6.109). We find that the solution to this minimax problem yields $F^* = 8660.2540$ (ft-lbf-sec), which is an improvement over the saturation limit $h_{max} = 10000$. However, to achieve this value the behavior of the other states and control become significantly less smooth as illustrated by the dotted lines in Figures 6.29–6.31.

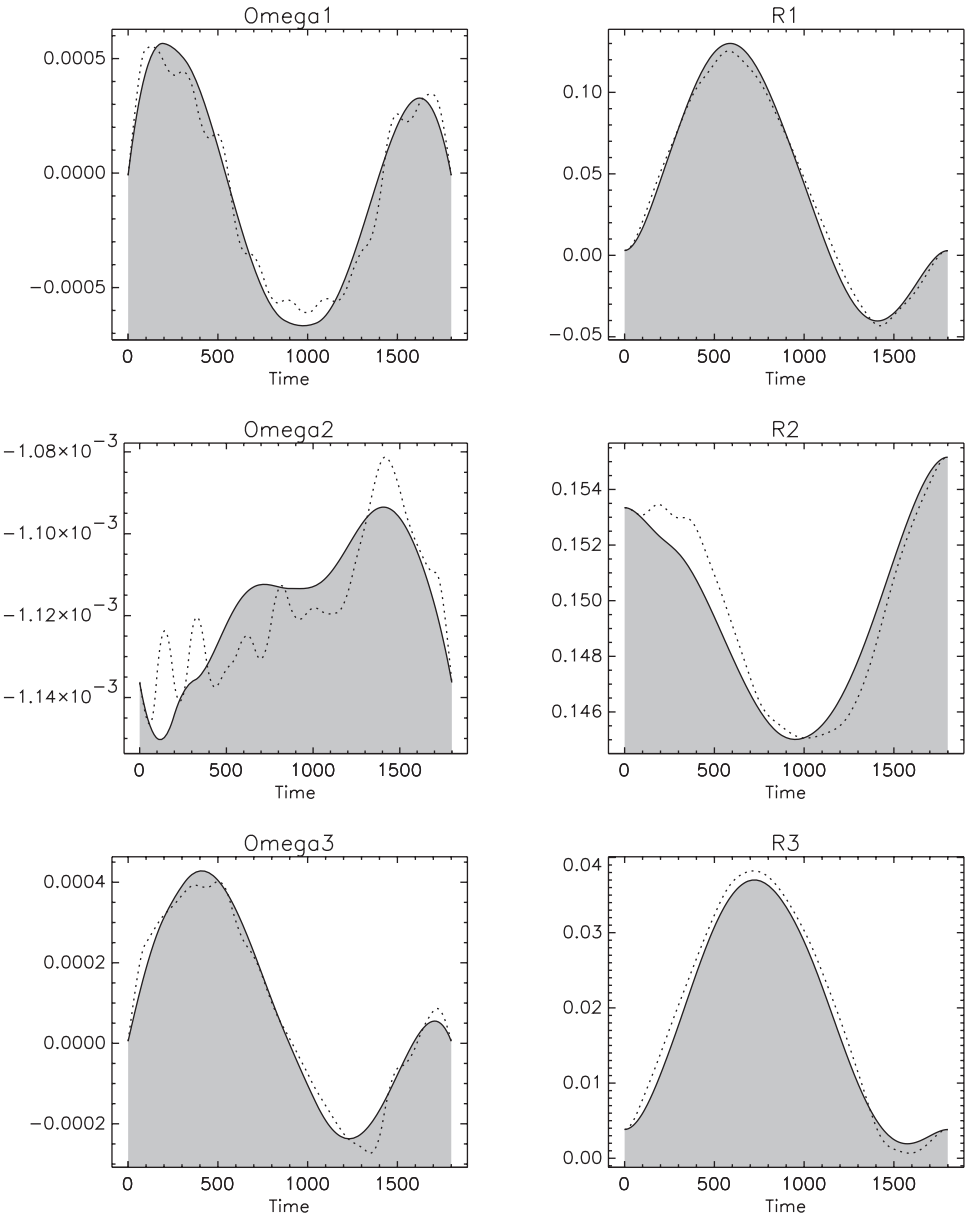


Figure 6.29. *ISS momentum dumping; states $\omega_k(t)$, $r_k(t)$.*

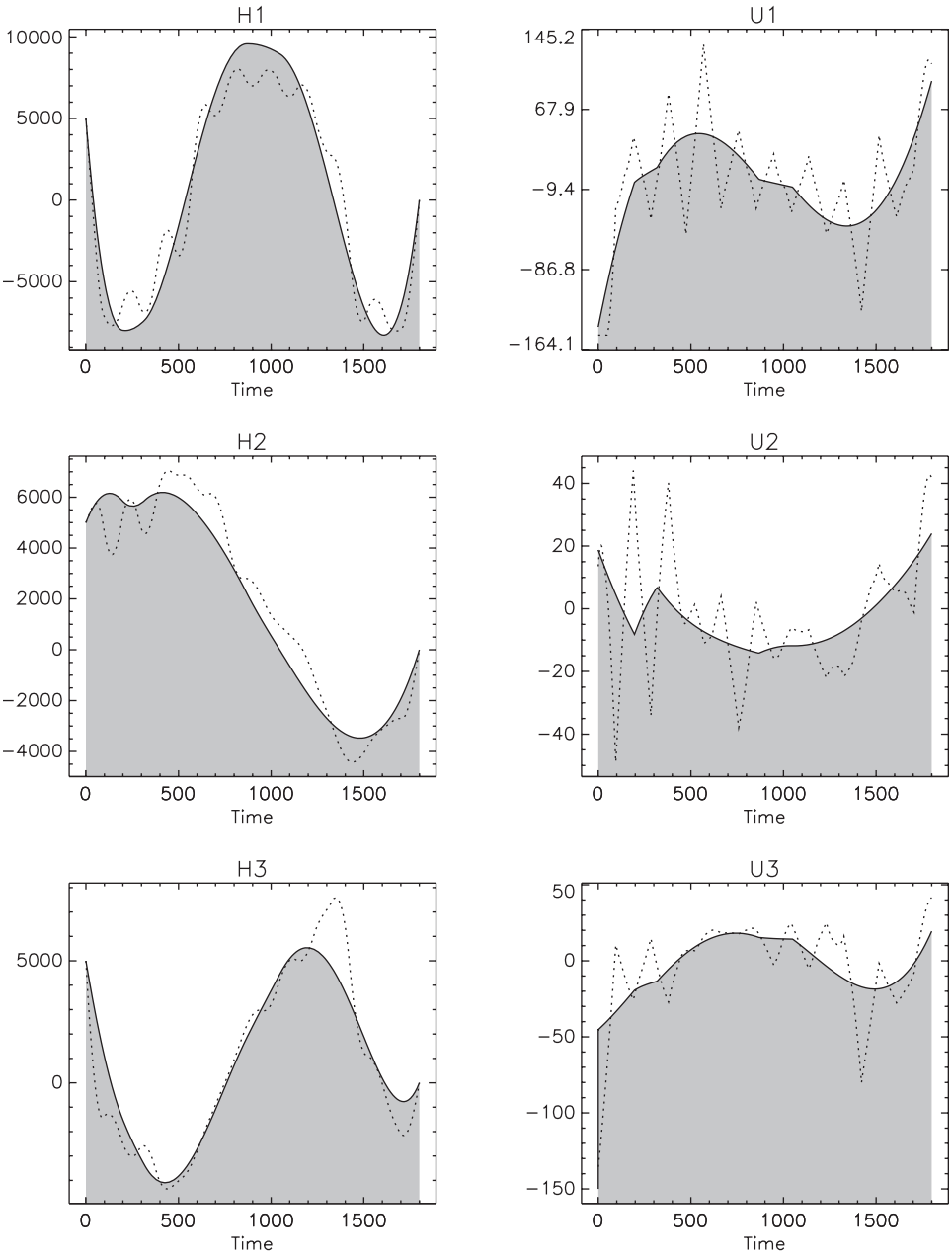


Figure 6.30. *ISS momentum dumping; states $h_k(t)$, controls $u_k(t)$.*

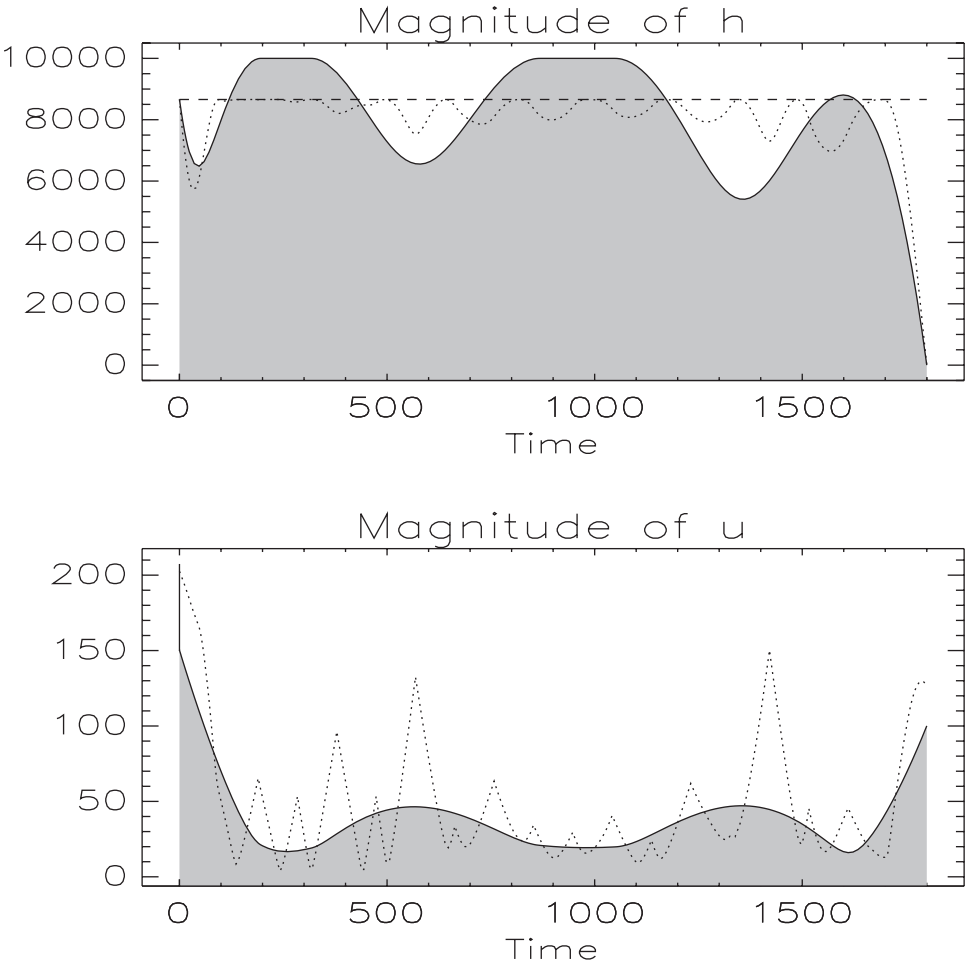


Figure 6.31. *ISS momentum dumping; magnitudes* $\|\mathbf{h}(t)\|$ *and* $\|\mathbf{u}(t)\|$.

Table 6.13. *Space station mesh-refinement summary.*

k	M	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	20	13	2	365	12537	2.0720×10^{-2}	9.0000×10^{-2}
2	39	6	2	270	20658	2.1309×10^{-3}	1.0000×10^{-1}
3	39	9	2	1170	113588	4.2938×10^{-5}	3.7000×10^{-1}
4	77	4	1	544	128606	9.0924×10^{-6}	4.5000×10^{-1}
5	87	4	1	544	149154	1.3774×10^{-6}	5.4000×10^{-1}
6	169	3	1	489	267935	5.1763×10^{-8}	1.0500×10^{0}
Total	169	39	9	3382	692478		2.6000×10^{0}

k Mesh-Refinement Number; M Number of Mesh Points
NGC Number of Gradient Calls; NHC Number of Hessian Calls
NFE Number of Function Evaluations; NRHS Number of Right-Hand-Side Evaluations
ϵ Discretization Error; Time (sec) CPU Time

6.8 Reorientation of an Asymmetric Rigid Body

Example 6.12 REORIENTATION OF RIGID BODY. Fleming, Sekhavat, and Ross [80] describe an application of optimal control techniques to the attitude control of a spacecraft. The dynamics are given by the following system of ODEs:

$$\dot{q}_1 = \frac{1}{2}\left[\omega_1 q_4 - \omega_2 q_3 + \omega_3 q_2\right], \tag{6.123a}$$

$$\dot{q}_2 = \frac{1}{2}\left[\omega_1 q_3 + \omega_2 q_4 - \omega_3 q_1\right], \tag{6.123b}$$

$$\dot{q}_3 = \frac{1}{2}\left[-\omega_1 q_2 + \omega_2 q_1 + \omega_3 q_4\right], \tag{6.123c}$$

$$\dot{q}_4 = \frac{1}{2}\left[-\omega_1 q_1 - \omega_2 q_2 - \omega_3 q_3\right], \tag{6.123d}$$

$$\dot{\omega}_1 = \frac{u_1}{I_x} - \left(\frac{I_z - I_y}{I_x}\right)\omega_2\omega_3, \tag{6.123e}$$

$$\dot{\omega}_2 = \frac{u_2}{I_y} - \left(\frac{I_x - I_z}{I_y}\right)\omega_1\omega_3, \tag{6.123f}$$

$$\dot{\omega}_3 = \frac{u_3}{I_z} - \left(\frac{I_y - I_x}{I_z}\right)\omega_1\omega_2. \tag{6.123g}$$

In this formulation the orientation is defined using the four-parameter set

$$\mathbf{q}^{\mathsf{T}} = (q_1, q_2, q_3, q_4) = (\widetilde{\mathbf{q}}, q_4), \tag{6.124}$$

called Euler parameters or quaternions [116, pp. 18–31], where $\|\mathbf{q}\| = 1$. The angular velocity is defined by the vector $\boldsymbol{\omega}^{\mathsf{T}} = (\omega_1, \omega_2, \omega_3)$. For their example, the authors model the behavior of the NASA X-ray Timing Explorer (XTE) spacecraft in which case the moments of inertia are given by

$$I_x = 5621 \quad \mathrm{Kg}\cdot\mathrm{m}^2, \qquad I_y = 4547 \quad \mathrm{Kg}\cdot\mathrm{m}^2, \qquad I_z = 2364 \quad \mathrm{Kg}\cdot\mathrm{m}^2$$

and the control torques $\mathbf{u} = (u_1, u_2, u_3)$ are limited by

$$\|\mathbf{u}\|_\infty \leq 50 \quad \mathrm{N}\cdot\mathrm{m}. \tag{6.125}$$

When using this formulation the differential variables are $\mathbf{y}^{\mathsf{T}} = (\mathbf{q}^{\mathsf{T}}, \boldsymbol{\omega}^{\mathsf{T}})$ and the algebraic variables are $\mathbf{u}$.

Since the quaternions must have norm one, an alternative is to omit differential equation (6.123d) from the ODE system (6.123a)–(6.123g) and replace it with an algebraic

constraint leading to the DAE system

$$\dot{q}_1 = \frac{1}{2}\left[\omega_1 q_4 - \omega_2 q_3 + \omega_3 q_2\right], \tag{6.126a}$$

$$\dot{q}_2 = \frac{1}{2}\left[\omega_1 q_3 + \omega_2 q_4 - \omega_3 q_1\right], \tag{6.126b}$$

$$\dot{q}_3 = \frac{1}{2}\left[-\omega_1 q_2 + \omega_2 q_1 + \omega_3 q_4\right], \tag{6.126c}$$

$$\dot{\omega}_1 = \frac{u_1}{I_x} - \left(\frac{I_z - I_y}{I_x}\right)\omega_2\omega_3, \tag{6.126d}$$

$$\dot{\omega}_2 = \frac{u_2}{I_y} - \left(\frac{I_x - I_z}{I_y}\right)\omega_1\omega_3, \tag{6.126e}$$

$$\dot{\omega}_3 = \frac{u_3}{I_z} - \left(\frac{I_y - I_x}{I_z}\right)\omega_1\omega_2, \tag{6.126f}$$

$$0 = \|\mathbf{q}\| - 1. \tag{6.126g}$$

When posed in this manner the differential variables are $\mathbf{y}^\mathsf{T} = (\widetilde{\mathbf{q}}^\mathsf{T}, \boldsymbol{\omega}^\mathsf{T})$, and the algebraic variables are $\widehat{\mathbf{u}} = (q_4, \mathbf{u})$. Observe that the quaternion q_4 is treated as an algebraic state. The goal is to reorient the spacecraft by executing a 150 deg roll maneuver about the x-axis in minimum time. The initial and final states are specified by the boundary conditions[5]

$$\mathbf{q}^\mathsf{T}(0) = (0,0,0,1), \quad \mathbf{q}^\mathsf{T}(t_F) = \left(\sin\frac{\phi}{2}, 0, 0, \cos\frac{\phi}{2}\right), \quad \boldsymbol{\omega}(0) = \mathbf{0}, \quad \boldsymbol{\omega}(t_F) = \mathbf{0}, \tag{6.127}$$

where $\phi = 150^\circ$ is the Euler axis rotation angle.

6.8.1 Computational Issues

As posed the problem illustrates a number of subtle computational issues. First, there are many local solutions to this problem all with the same value of the optimal time, namely $t_F^* = 28.630408$ (sec). Two typical solutions are plotted in Figures 6.32–6.34, and numerical experimentation suggests there are other solutions as well. Since the controls appear linearly in the problem, the solution is "bang-bang." However, the optimality conditions do not determine the particular *sequence* of boundary arcs. For example, notice that in Figure 6.34 the solution plotted as a solid line begins with $\mathbf{u}(0) = (+50, -50, +50)$. In contrast the second solution plotted with a dotted line begins with $\mathbf{u}(0) = (+50, -50, -50)$. The controls must be on either the upper or lower bound in order to satisfy the maximum principle; however, there is no restriction on *which bound!* Clearly, different combinations can lead to different arc sequences, which are all potential local optimizers.

Second, it is not clear what (if any) difference there is between the ODE formulation (6.123a)–(6.123g) and the DAE formulation (6.126a)–(6.126g). To address this let us consider the computational behavior on one particular case, in which we (arbitrarily) add the additional boundary conditions $\mathbf{u}^\mathsf{T}(0) = (+50, -50, +50)$ to the required conditions (6.127). For this experiment let us solve the problem using the standard mesh-refinement

[5] Boundary values correct a typographical error in [80].

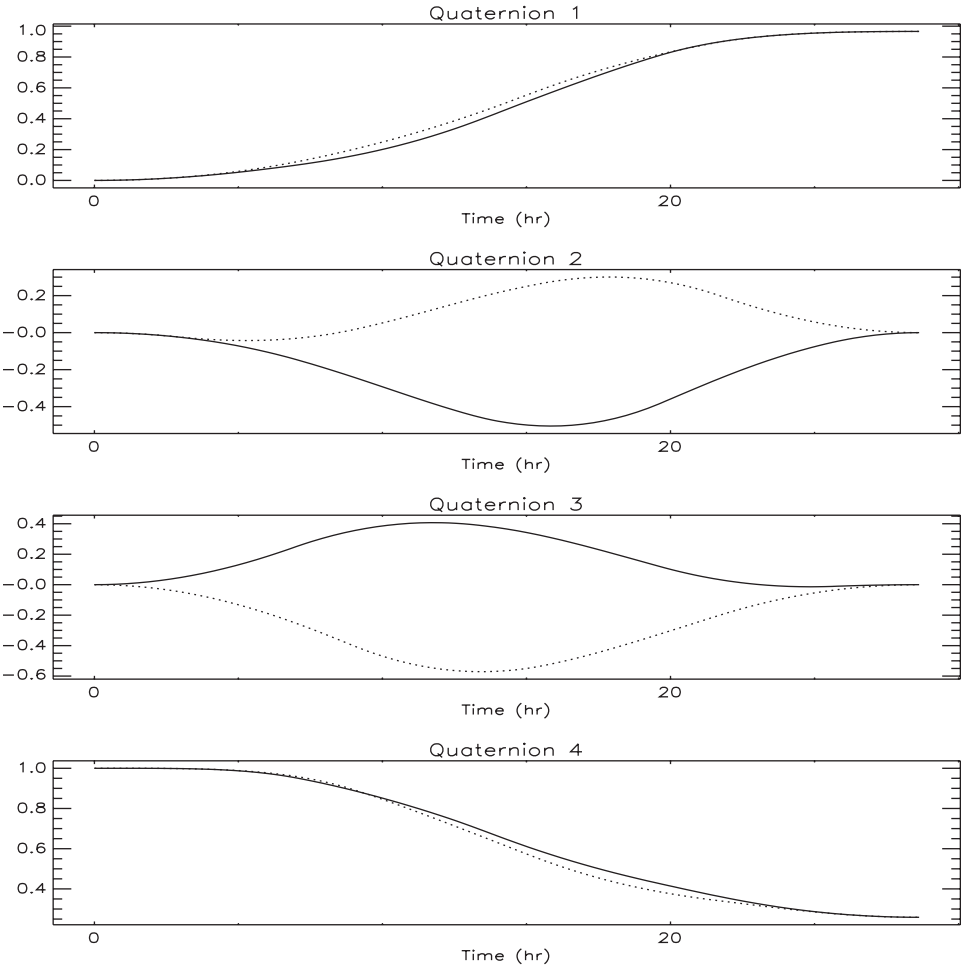


Figure 6.32. *Optimal quaternion history.*

procedure, beginning with a trapezoidal discretization, and a mesh-refinement tolerance of $\epsilon_{\max} = 10^{-7}$. As an initial guess we use a straight line between the initial and final conditions (with $\mathbf{u}^{(0)}(t_F) = \mathbf{0}$) and a guessed value $t_F^{(0)} = 30$. Table 6.14 summarizes the results of six different cases. For the first four cases the initial grid consisted of 20 equally distributed points, and the final two cases utilize an initial grid with 50 points. Cases 1 and 2 summarize the results using the DAE problem formulation with the SQP and Barrier NLP algorithms, respectively. The computational results strongly suggest that the DAE problem is superior. When the problem is formulated as an ODE (using (6.123a)–(6.123g)) we observe either ill-conditioning and/or slow convergence, regardless of the choice of NLP algorithm. The difficulty can be attributed to a fundamental numerical problem associated with quaternions. The DAE formulation explicitly ensures satisfaction of the normalization condition $\|\mathbf{q}(t)\| = 1$ at every grid point, even when the mesh is coarse. In contrast to en-

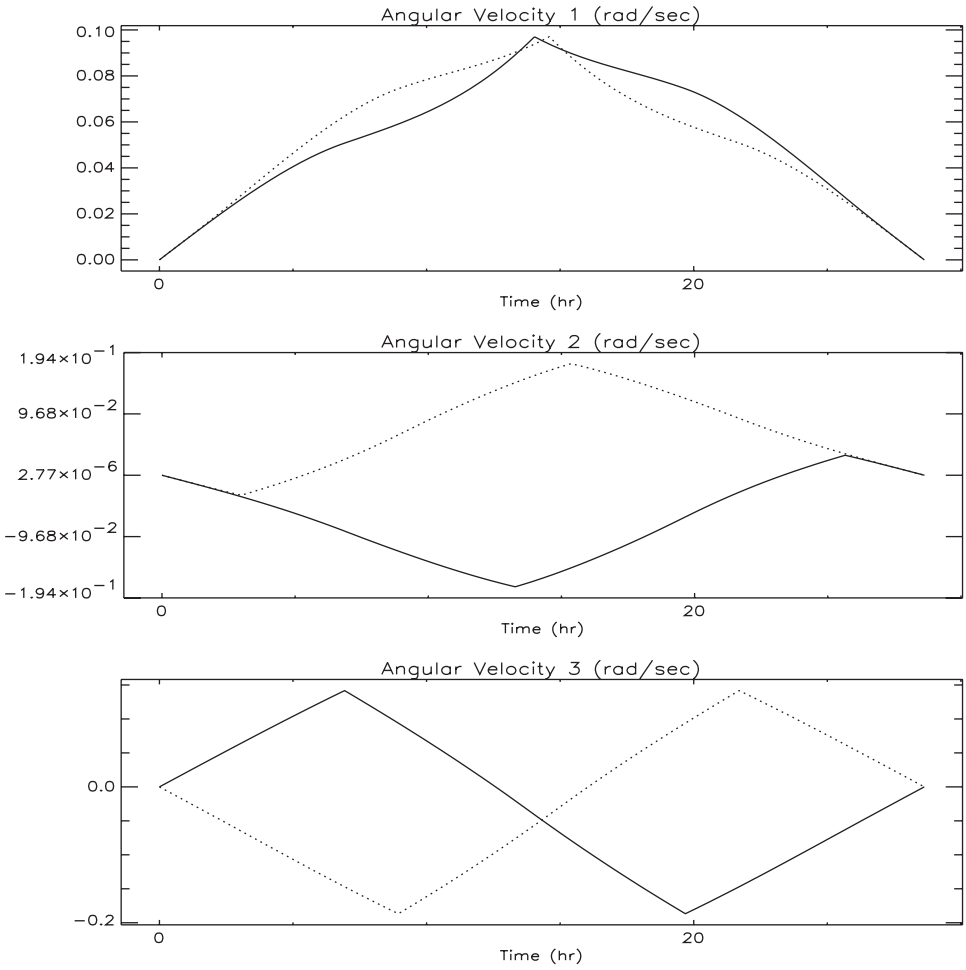


Figure 6.33. *Optimal velocity history.*

sure the normalization holds when using the ODE formulation, it is necessary to integrate the ODE *very* accurately. In other words, one cannot expect $\|\mathbf{q}(t)\| = 1$ unless the ODEs (6.123a)–(6.123d) are solved accurately. In particular, even if $\|\mathbf{q}(0)\| = 1$, we must expect that integration error will cause $\|\mathbf{q}(t_F)\| \neq 1$, thereby leading to an ill-conditioned BVP.

It is instructive to examine the behavior of the normalization condition $\|\mathbf{q}\| = 1$ after a single step. Specifically consider a single trapezoidal step

$$0 = \mathbf{y}_1 - \mathbf{y}_0 - \frac{h}{2}(\mathbf{f}_0 + \mathbf{f}_1), \tag{6.128}$$

where $\mathbf{y}^{\mathsf{T}} = (\mathbf{q}^{\mathsf{T}}, \boldsymbol{\omega}^{\mathsf{T}})$ and $\mathbf{f}$ is defined by the right-hand sides of the ODE system (6.123a)–(6.123g). Let us assume $\mathbf{y}_0^{\mathsf{T}} = (0,0,0,1,0,0,0)$ as in (6.127) and further assume

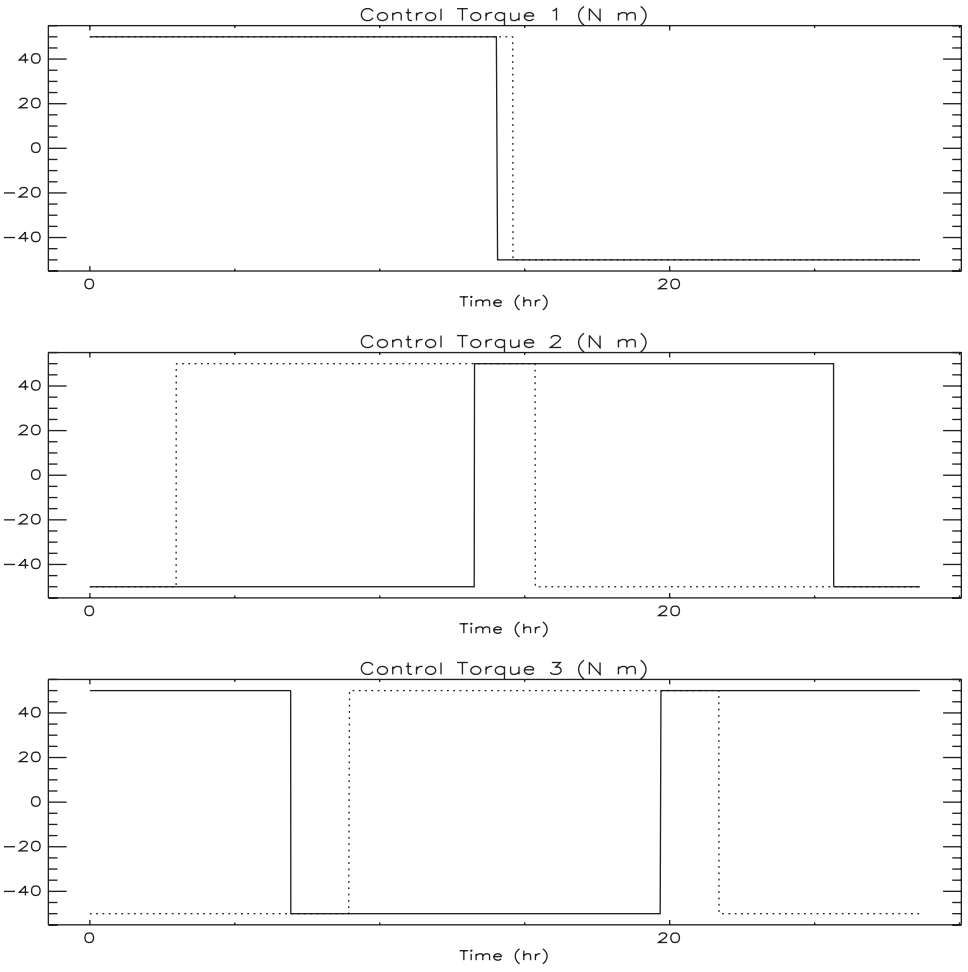


Figure 6.34. *Optimal control torque history.*

$\mathbf{u}^{\mathsf{T}} = (+50, -50, +50)$ over the entire step. Clearly $\|\mathbf{q}(0)\| = 1$. Figure 6.35 illustrates the normalization error in the first four components of $\mathbf{y}$ for a range of stepsizes. Clearly, the numerical integration does not preserve normalization *even over a single step.* Unfortunately, this result has catastrophic implications when present in the boundary value context. In particular, if the (correctly defined) boundary conditions require $\|\mathbf{q}(0)\| = 1$ and $\|\mathbf{q}(h)\| = 1$, this cannot be achieved unless the numerical integration scheme has zero error! As a consequence when a direct transcription method is used to solve the BVP, the constraints are degenerate, leading to a rank-deficient Jacobian matrix, and the overall success or failure of the method will depend on how the NLP treats rank deficiency. Since this difficulty is not unique to the trapezoidal discretization, or, for that matter, the direct transcription method, it is preferable to simply pose the problem as a DAE.

Table 6.14. *Performance comparison.*

Formulation	NLP	Ref. Iter.	Grid Pts.	Func. Eval.	Time (sec)
DAE	SQP	9	136	10572	25.01
DAE	Barrier	9	130	37407	19.52
ODE	SQP	8[a]	353	28999	121.9
ODE	Barrier	1[b]	20	3243	.58
ODE	SQP	9[c]	349	17340	131.02
ODE	Barrier	1[d]	50	3597	1.56

[a] Abnormal Termination (Incorrect Inertia Ill-Conditioning
[b] Abnormal Termination (Ill-Conditioning) Final Objective $t_F^* = 28.877792$.
[c] Optimal Objective $t_F^* = 28.661727$.
[d] Abnormal Termination; Final Objective $t_F^* = 28.664134$.

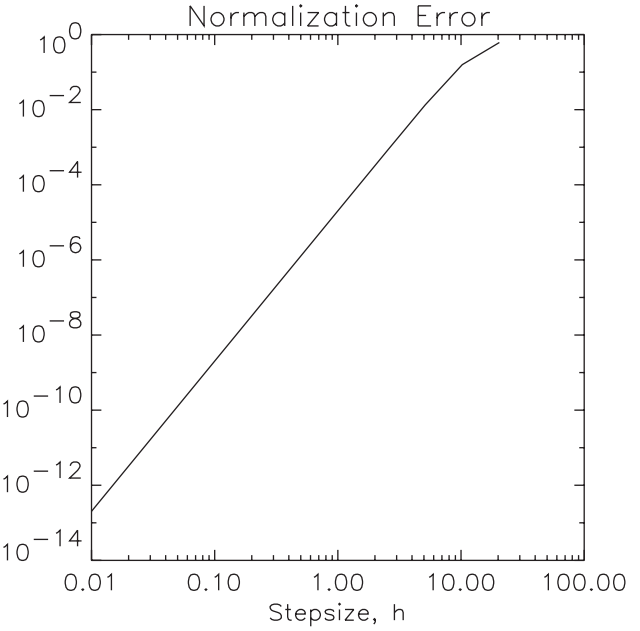


Figure 6.35. *Stepwise normalization error growth* $1 - \|\mathbf{q}(h)\|$.

6.9 Industrial Robot

Example 6.13 INDUSTRIAL ROBOT. An interesting example describing the motion of an industrial robot is presented in [169]. The machine is shown in Figure 6.36 and is called the Manutec r3 [138]. It has six degrees of freedom, although only three degrees of freedom are considered in this formulation since they adequately describe the position of the tool center point. The dynamics are described by the second order system

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} = \mathbf{f}(\dot{\mathbf{q}}, \mathbf{q}, \mathbf{u}), \tag{6.129}$$

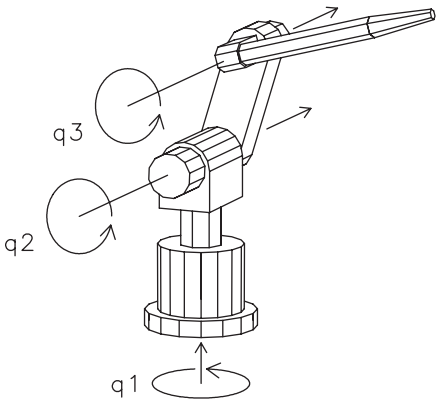


Figure 6.36. *Manutec r*3 *robot.*

where the vector $\mathbf{q} = [q_1(t), q_2(t), q_3(t)]^{\mathsf{T}}$ defines the relative angles between the robot arms. The torque voltages $\mathbf{u} = [u_1(t), u_2(t), u_3(t)]^{\mathsf{T}}$ are applied to control the robot motors. The symmetric positive definite mass matrix $\mathbf{M}(\mathbf{q})$ involves very complicated expressions of the state $\mathbf{q}$, as does the function $\mathbf{f}$, which defines the moments caused by Coriolis, centrifugal, and gravitational forces. A more complete description of the quantities in this *multibody system* can be found in [169]. If the angular velocities are denoted by $\mathbf{v}$, then (6.129) can be written as the first order system

$$\dot{\mathbf{q}} = \mathbf{v}, \tag{6.130}$$

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{f}(\mathbf{v}, \mathbf{q}, \mathbf{u}). \tag{6.131}$$

Because the matrix $\mathbf{M}(\mathbf{q})$ has a special tree structure, it is possible to construct the inverse using an $\mathcal{O}(n)$ algorithm ($n = 3$), as described in [138]. Thus, we obtain the semiexplicit first order system

$$\dot{\mathbf{q}} = \mathbf{v}, \tag{6.132}$$

$$\dot{\mathbf{v}} = \mathbf{M}^{-1}(\mathbf{q})\mathbf{f}(\mathbf{v}, \mathbf{q}, \mathbf{u}). \tag{6.133}$$

It should be emphasized that the right-hand side of (6.133) involves *very complicated* expressions that are often generated automatically by simulation software systems. The computational results use software graciously provided by O. von Stryk to compute these differential equations.

The goal is to construct a trajectory for the robot tool tip that goes from one specified location to another (a "point-to-point" path). Also, we want the tool to begin and end the trajectory at rest, i.e., with zero velocity. Thus, we have the boundary conditions

$$\mathbf{q}(0) = (0, -1.5, 0)^{\mathsf{T}} \text{ (rad)}, \qquad \mathbf{v}(0) = \mathbf{0} \text{ (rad/sec)},$$

$$\mathbf{q}(t_F) = (1, -1.95, 1)^{\mathsf{T}} \text{ (rad)}, \qquad \mathbf{v}(t_F) = \mathbf{0} \text{ (rad/sec)}.$$

This particular tool also has physical limits that restrict the state and control variables.

Specifically,

$$\begin{aligned}
\|\mathbf{u}(t)\|_\infty &\le 7.5 \text{ (V)},\\
|q_1(t)| &\le 2.97 \text{ (rad)},\\
|q_2(t)| &\le 2.01 \text{ (rad)},\\
|q_3(t)| &\le 2.86 \text{ (rad)},\\
|v_1(t)| &\le 3.0 \text{ (rad/sec)},\\
|v_2(t)| &\le 1.5 \text{ (rad/sec)},\\
|v_3(t)| &\le 5.2 \text{ (rad/sec)}.
\end{aligned}$$

Three different objective functions are considered in [169], the first two being minimum time

$$J = t_F \tag{6.134}$$

and minimum energy

$$J = \int_0^{t_F} \mathbf{u}^\mathsf{T}(t)\mathbf{u}(t)dt \tag{6.135}$$

for a fixed final time t_F.

The minimum energy problem is relatively straightforward and of significant practical value. However, a number of the concepts described in this book are illustrated by the minimum time problem, and so it will be the primary focus of the discussion. This example is useful because, as the authors suggest in their abstract,

> The highly accurate solutions reported in this paper may also serve as benchmark problems for other methods.

What is most admirable is *how* the authors obtained the solutions using an indirect multiple shooting method. As stated in [169],

> The main drawbacks when applying the multiple shooting method in the numerical solution of optimal control problems are, 1. the derivation of the necessary conditions (e.g., the adjoint differential equations), 2. the estimation of the optimal switching structure, and 3. the estimation of an appropriate initial guess of the unknown state and adjoint variables $x(t), \lambda(t), \eta(t)$ in order to start the iteration process.

Briefly, their approach was to derive the adjoint equations using the software tool Maple, which produced over 4000 lines of FORTRAN code. Then a direct transcription method (with a coarse mesh) was used to construct both an initial guess for the switching structure and estimates for the state and adjoint variables. Finally, this information was used to initialize an indirect multiple shooting method. Our goal here is to solve the same problem *without* deriving the adjoint equations and guessing the adjoint variables.

As posed, the minimum time problem presents two major difficulties. First, the control variable appears linearly in the differential equations. Consequently, it is most likely that the control will be bang-bang, as discussed in Example 4.11. Singular subarcs, as described in Example 4.9, are also possible. However, for the specific boundary conditions given here, they do not appear. A more complete analysis can be found in [169]. Second, when the limits on the angular velocity are active, the resulting DAE has index two. Since

the Jacobian matrix is singular on the state constraints, the control variable is not well defined by the usual necessary conditions, as illustrated by Example 4.10. The approach we will follow is to first estimate the switching structure and then solve a multiphase problem with index reduction on the phases corresponding to the active state boundaries.

The first step is to compute an estimate for the switching structure. To do this, let us solve a modified version of the minimum time problem. Specifically, minimize

$$J = t_F + \rho \int_0^{t_F} \mathbf{u}^{\mathsf{T}}(t)\mathbf{u}(t)dt, \tag{6.136}$$

where the "small" parameter $\rho = 1 \times 10^{-5}$ is chosen to regularize the problem. By introducing this quadratic regularization, the control becomes uniquely defined and the solution should be "close" to the minimum time problem. It is not necessary (or desirable) to solve this modified problem to a high degree of precision because the primary goal is to construct the appropriate switching structure. Table 6.15 summarizes the results, which were intentionally terminated after seven mesh-refinement iterations. In fact, the computed approximation to the minimum time is 0.495196288 sec, which agrees with the true optimum value 0.49518904 sec to four significant figures. Although there is very little "visible" difference in the objective function, there is a visible difference in the control history. It is interesting to note that the mesh-refinement algorithm presented in Section 4.6.9 selected the separated Simpson discretization because the function evaluations were relatively expensive.

Table 6.15. *Minimum time with regularization.*

Iter.	Disc.	M	GE	HE	FE	RHS	$\epsilon_{\max}$	CPU (sec)
1	TR	10	16	10	619	6190	0.11×10^{-1}	0.27×10^{1}
2	TR	19	10	8	469	8911	0.86×10^{-2}	0.44×10^{1}
3	HS	19	7	5	304	11248	0.23×10^{-2}	0.14×10^{2}
4	HS	37	7	5	304	22192	0.20×10^{-2}	0.37×10^{2}
5	HS	46	8	6	359	32669	0.36×10^{-3}	0.57×10^{2}
6	HS	84	7	5	304	50768	0.50×10^{-4}	0.19×10^{3}
7	HS	97	5	3	194	37442	0.13×10^{-4}	0.19×10^{3}
Total		97	60	42	2553	169420		493.53

The solution to the relaxed problem is illustrated by the dashed line in Figure 6.37. The relative accuracy of the solution is dominated by the discretization error in the neighborhood of the discontinuities. However, instead of trying to improve the accuracy by adding grid points, it is better to explicitly introduce this discontinuous behavior using separate phases. Examination of the solution provides an estimate for the switching structure. Thus, we are led to model the behavior using nine distinct phases with each phase characterized by a different set of active path constraints. Table 6.16 summarizes the phase-by-phase situation. Thus, in phase 1, the first three angular velocities v_1, v_2, and v_3 are free, whereas the first two control variables u_1 and u_2 are at their minimum values, while the final control u_3 is a maximum.

To illustrate how the multiple-phase structure is exploited, let us consider the dynamics in phase 3. In phase 3, state variable v_2 is on its lower bound and state variable v_3 is on

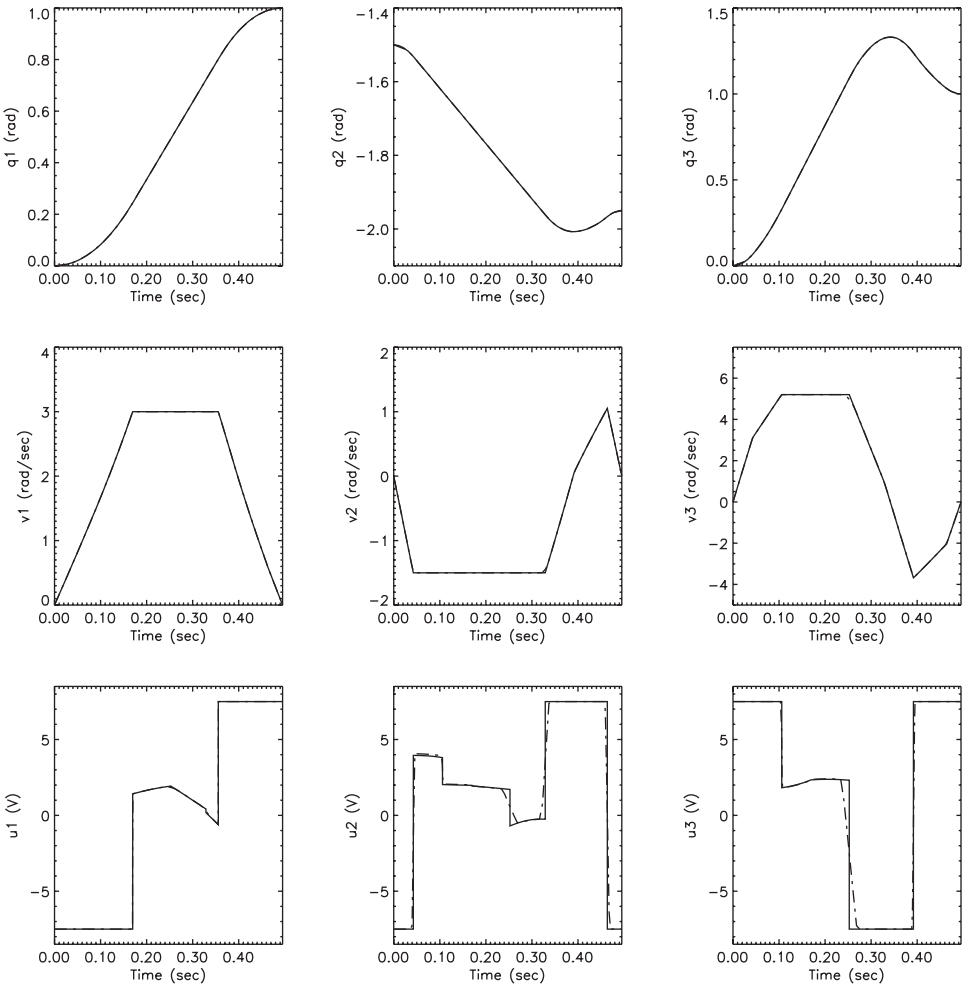


Figure 6.37. *Robot solution.*

Table 6.16. *Minimum time switching structure.*

Phase	v_1	v_2	v_3	u_1	u_2	u_3
1	free	free	free	min	min	max
2	free	min	free	min	free	max
3	free	min	max	min	free	free
4	max	min	max	free	free	free
5	max	min	free	free	free	min
6	max	free	free	free	max	min
7	free	free	free	max	max	min
8	free	free	free	max	max	max
9	free	free	free	max	min	max

its upper bound. Thus, we must impose the algebraic constraints

$$0 = v_2(t) - v_{2L}, \tag{6.137}$$
$$0 = v_3(t) - v_{3U}, \tag{6.138}$$

where the lower bound $v_{2L} = -1.5$ and the upper bound $v_{3U} = 5.2$. Neither (6.137) nor (6.138) explicitly contains a control variable and, as such, both are index-two path constraints. To reduce the index, we must differentiate with respect to time giving

$$0 = \dot{v}_2(t), \tag{6.139}$$
$$0 = \dot{v}_3(t). \tag{6.140}$$

Thus, from (6.132) and (6.133) we must have

$$\dot{q}_1 = v_1(t), \tag{6.141}$$
$$\dot{q}_2 = v_{2L}, \tag{6.142}$$
$$\dot{q}_3 = v_{3U}, \tag{6.143}$$
$$\dot{v}_1 = \left\{\mathbf{M}^{-1}(\mathbf{q})\mathbf{f}(\mathbf{v},\mathbf{q},\mathbf{u})\right\}_1, \tag{6.144}$$
$$0 = \left\{\mathbf{M}^{-1}(\mathbf{q})\mathbf{f}(\mathbf{v},\mathbf{q},\mathbf{u})\right\}_2, \tag{6.145}$$
$$0 = \left\{\mathbf{M}^{-1}(\mathbf{q})\mathbf{f}(\mathbf{v},\mathbf{q},\mathbf{u})\right\}_3. \tag{6.146}$$

Since $\mathbf{v} = (v_1(t), v_{2L}, v_{3U})$ and $\mathbf{u} = (u_{1L}, u_2(t), u_3(t))$, the dynamics in phase 3 are completely defined by this (index-one) DAE system. Thus, phase 3 can be modeled using four (not six) differential equations and two (not three) algebraic equations. The differential (state) variables in this phase are

$$(q_1(t), q_2(t), q_3(t), v_1(t))$$

and the algebraic (control) variables are $(u_2(t), u_3(t))$. The beginning and end of the phase are not specified, so we must treat the values $t_I^{(3)}$ and $t_F^{(3)}$ as additional NLP variables, where phase 3 is defined on the domain $t_I^{(3)} \le t \le t_F^{(3)}$. The phase description is complete when boundary conditions are specified. Continuity at the beginning and end of the phase is enforced by imposing the boundary conditions

$$\begin{aligned} t_F^{(2)} &= t_I^{(3)}, & t_F^{(3)} &= t_I^{(4)}, \\ q_1(t_F^{(2)}) &= q_1(t_I^{(3)}), & q_1(t_F^{(3)}) &= q_1(t_I^{(4)}), \\ q_2(t_F^{(2)}) &= q_2(t_I^{(3)}), & q_2(t_F^{(3)}) &= q_2(t_I^{(4)}), \\ q_3(t_F^{(2)}) &= q_3(t_I^{(3)}), & q_3(t_F^{(3)}) &= q_3(t_I^{(4)}), \\ v_1(t_F^{(2)}) &= v_1(t_I^{(3)}), & v_1(t_F^{(3)}) &= v_1(t_I^{(4)}). \end{aligned}$$

To ensure the proper transition for the state-constrained variables between phases 2, 3, and 4, at the end of phase 2 we must impose the condition $v_3(t_F^{(2)}) = v_{3U}$ and, at the end of phase 3, it is necessary that $v_1(t_F^{(3)}) = v_{1U}$. Observe that no continuity conditions are imposed on

the control variables. In fact, one benefit of the multiphase treatment of this problem is that it does permit accurate modeling of the control discontinuities.

It is worth noting that the technique presented here models phase 3 with a reduced set of state and control variables. Computationally, this is advantageous because the size of the transcribed problem has been reduced. However, it may be more convenient to implement a problem that has the same number of states and controls on each phase. This can be achieved by adding the required path equality constraints on each phase and then simply solving a larger NLP problem. This method is outlined in the doctoral thesis of O. von Stryk.

For brevity, we omit an explicit description of the approach for all nine phases. However, it should be clear that with a known switching structure it is relatively straightforward to implement the correct conditions in a computational tool such as SOCS. When this information is explicitly incorporated into a nine-phase formulation, the resulting nonlinear program is well-posed and leads to the solution illustrated by the solid line in Figure 6.37. Table 6.17, which summarizes the behavior of the algorithm, clearly demonstrates the benefit of incorporating the switching structure into the formulation.

Table 6.17. *Minimum time with switching structure.*

Iter.	Disc.	M	GE	HE	FE	RHS	$\epsilon_{\max}$	CPU (sec)
1	TR	45	4	0	34	1530	0.18×10^{-4}	0.34×10^{1}
2	HC	45	2	0	34	2754	0.25×10^{-6}	0.22×10^{1}
3	HC	57	1	0	18	1890	0.66×10^{-7}	0.23×10^{1}
Total		57	7	0	86	6174		7.81

As a final point of interest, it is worth comparing the accuracy of the solutions computed using the direct transcription approach with those obtained by the benchmark indirect multiple shooting approach [169]. The optimal objective function values are given in Table 6.18 for both methods, with the significant figures that agree underlined. The results agree to seven significant figures in both cases, which also validates the reliability of the discretization error $\epsilon_{\max} \leq 1 \times 10^{-7}$. More important is the fact that the direct transcription results were obtained without computing the adjoint equations and estimating values for the adjoint variables. This is especially encouraging since it suggests that accurate results can be obtained for many practical applications even when it is too complicated to form the adjoint equations.

Table 6.18. *Accuracy comparison.*

Problem	Indirect multiple shooting	Direct transcription
Min. energy	20.404247	20.40424581
Min. time	0.49518904	0.495189037

6.10 Multibody Mechanism

Example 6.14 ANDREW'S SQUEEZER MECHANISM. Hairer and Wanner [107, pp. 530–542] describe a very nice example of a *multibody system* called "Andrew's squeezer

mechanism" and have graciously supplied a software implementation of the relevant equations. The problem is used as a benchmark for testing a number of different multibody simulation codes as described in [154]. For the sake of brevity, we omit a detailed description of the problem and refer the reader to [107]. The multibody dynamics are described by a second order system similar to (6.129):

$$\mathbf{M}(\mathbf{p})\ddot{\mathbf{p}} = \mathbf{f}(\dot{\mathbf{p}}, \mathbf{p}, u) - \mathbf{G}^{\mathsf{T}}(\mathbf{p})\boldsymbol{\lambda}, \tag{6.147}$$

$$\mathbf{0} = \mathbf{g}(\mathbf{p}). \tag{6.148}$$

For this example, the "position" vector $\mathbf{p} = (\beta, \Theta, \gamma, \Phi, \delta, \Omega, \varepsilon)^{\mathsf{T}}$ consists of seven angles that define the orientation of the seven-body mechanism. The bodies are linked together via the algebraic constraints (6.148). The symmetric *mass matrix* $\mathbf{M}$ is tridiagonal, and the derivation of these equations is often facilitated by noting that

$$m_{ij} = \frac{\partial^2 T}{\partial \dot{p}_i \partial \dot{p}_j},$$

where T is the kinetic energy of the system. The mechanism is driven by a motor whose drive torque is given by u. In [107], the torque $u_0 = 0.033$ Nm is treated as a constant, in which case this is simply an IVP. For the sake of illustration, let us assume that the drive torque is a control variable that can be adjusted as a function of time. Let us impose bounds

$$0 \le u(t) \le 0.066 \qquad (= 2 \times 0.033) \tag{6.149}$$

and then try to compute the control such that

$$J = \frac{1}{t_F u_0^2} \int_0^{t_F} u^2(t)dt \tag{6.150}$$

is minimized over the time domain $0 \le t \le t_F = 0.03$ ms.

First, let us convert (6.147) from a second order implicit form to a first order semiexplicit DAE system giving

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{6.151}$$

$$\dot{\mathbf{v}} = \mathbf{q}, \tag{6.152}$$

$$\mathbf{0} = \mathbf{M}(\mathbf{p})\mathbf{q} - \mathbf{f}(\mathbf{v}, \mathbf{p}, u) + \mathbf{G}^{\mathsf{T}}(\mathbf{p})\boldsymbol{\lambda}, \tag{6.153}$$

$$\mathbf{0} = \mathbf{g}(\mathbf{p}). \tag{6.154}$$

In this formulation, $\mathbf{q}$, $\boldsymbol{\lambda}$, and u are the algebraic variables $\mathbf{u}$, whereas $\mathbf{p}$ and $\mathbf{v}$ are the differential variables $\mathbf{y}$. Observe that the complete set of algebraic variables denoted by $\mathbf{u}$ includes the "real" control u as well as the other algebraic variables $\mathbf{q}$ and $\boldsymbol{\lambda}$. If we differentiate (6.154) once with respect to time, we find

$$\mathbf{0} = \frac{d}{dt}\big[\mathbf{g}(\mathbf{p})\big] = \mathbf{G}(\mathbf{p})\dot{\mathbf{p}} = \mathbf{G}(\mathbf{p})\mathbf{v}. \tag{6.155}$$

A second differentiation with respect to time yields

$$\mathbf{0} = \frac{d^2}{dt^2}\big[\mathbf{g}(\mathbf{p})\big] = \frac{d}{dt}\big[\mathbf{G}(\mathbf{p})\mathbf{v}\big] = \dot{\mathbf{G}}(\mathbf{p})\mathbf{v} + \mathbf{G}(\mathbf{p})\dot{\mathbf{v}} = \mathbf{g}_{pp}(\mathbf{p})(\mathbf{v}, \mathbf{v}) + \mathbf{G}(\mathbf{p})\mathbf{q}. \tag{6.156}$$

Since the algebraic variable $\mathbf{q}$ appears in this *acceleration-level* constraint, no additional derivatives are needed. Observe that in its original form the problem is index three, and this illustrates the process of *index reduction*. Collecting results yields the index-one DAE system

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{6.157}$$

$$\dot{\mathbf{v}} = \mathbf{q}, \tag{6.158}$$

$$\mathbf{0} = \mathbf{M}(\mathbf{p})\mathbf{q} - \mathbf{f}(\mathbf{v}, \mathbf{p}, u) + \mathbf{G}^{\mathsf{T}}(\mathbf{p})\boldsymbol{\lambda}, \tag{6.159}$$

$$\mathbf{0} = \mathbf{g}_{pp}(\mathbf{p})(\mathbf{v}, \mathbf{v}) + \mathbf{G}(\mathbf{p})\mathbf{q}. \tag{6.160}$$

Since the original problem was index three, some care must be exercised when defining initial conditions to ensure that they are consistent. First, the initial conditions must satisfy (6.154), so following [107] we choose one of the position variables $\Theta(0) = 0$ and then compute the remaining six such that (6.154) is satisfied. The velocity-level constraint $\mathbf{0} = \mathbf{G}(\mathbf{p})\mathbf{v}$ is satisfied if we put $\mathbf{v}(0) = \mathbf{0}$. Finally, it remains to specify initial values for the algebraic variables $\mathbf{q}$, $\boldsymbol{\lambda}$, and u such that (6.159) and (6.160) hold. Rewriting these equations gives

$$\begin{bmatrix} \mathbf{M}(\mathbf{p}) & \mathbf{G}^{\mathsf{T}}(\mathbf{p}) \\ \mathbf{G}(\mathbf{p}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{v}, \mathbf{p}, u) \\ -\mathbf{g}_{pp}(\mathbf{p})(\mathbf{v}, \mathbf{v}) \end{bmatrix}. \tag{6.161}$$

Thus, for given values of the position $\mathbf{p}$, velocity $\mathbf{v}$, and control u, we can solve for the corresponding values of $\mathbf{q}$ and $\boldsymbol{\lambda}$. The initial values computed this way are consistent and *for a constant u* are sufficient to completely determine the solution to the IVP. However, when the torque u is allowed to vary with time, there are many solutions, and so, for comparison, we will impose the boundary condition $\beta(t_F) = p_1(t_F) = 15.8106$ rad. This specified value for the angle β is the same as the final value achieved when a constant torque is used. In other words, we will match the final state for the constant-torque IVP solution with the optimal control solution. Thus, the goal of the optimal control is to reach the same state in the same time while expending less "energy" (6.150).

The direct transcription method requires an initial guess for the state and control time functions. For most applications (including most other examples in this book), it suffices to supply a linear initial guess. In fact, the default procedure used by $\mathbb{SOCS}$ to compute the state at grid point k is

$$\mathbf{y}_k = \mathbf{y}_1 + \frac{(k-1)}{(M-1)}(\mathbf{y}_M - \mathbf{y}_1). \tag{6.162}$$

The user must specify the state at the initial grid point, $\mathbf{y}_1$, the state at the final grid point, $\mathbf{y}_M$, the initial and final times, t_I and t_F, and the number of grid points, M. A similar expression is used to define the control $\mathbf{u}_k$ and the grid time values t_k. However, it is often possible to construct a much better initial guess for the state and control variables using special information about the problem and, indeed, this is so for this multibody example. One obvious approach is to fix $u(t) = u_0$ and then integrate the index-one DAE system (6.157)–(6.160) using software for solving a DAE IVP such as DASSL [140, 141] or RADAU5 [107]. The second approach (also tested in [107]) is to apply an ODE method to the differential equations (6.157), (6.158). This technique requires an explicit expression for the vector $\mathbf{q}$ appearing on the right-hand side of (6.158). Fortunately, by solving (6.161),

$$\begin{bmatrix} \mathbf{q} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{M}(\mathbf{p}) & \mathbf{G}^{\mathsf{T}}(\mathbf{p}) \\ \mathbf{G}(\mathbf{p}) & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}(\mathbf{v}, \mathbf{p}, u_0) \\ -\mathbf{g}_{pp}(\mathbf{p})(\mathbf{v}, \mathbf{v}) \end{bmatrix}, \tag{6.163}$$

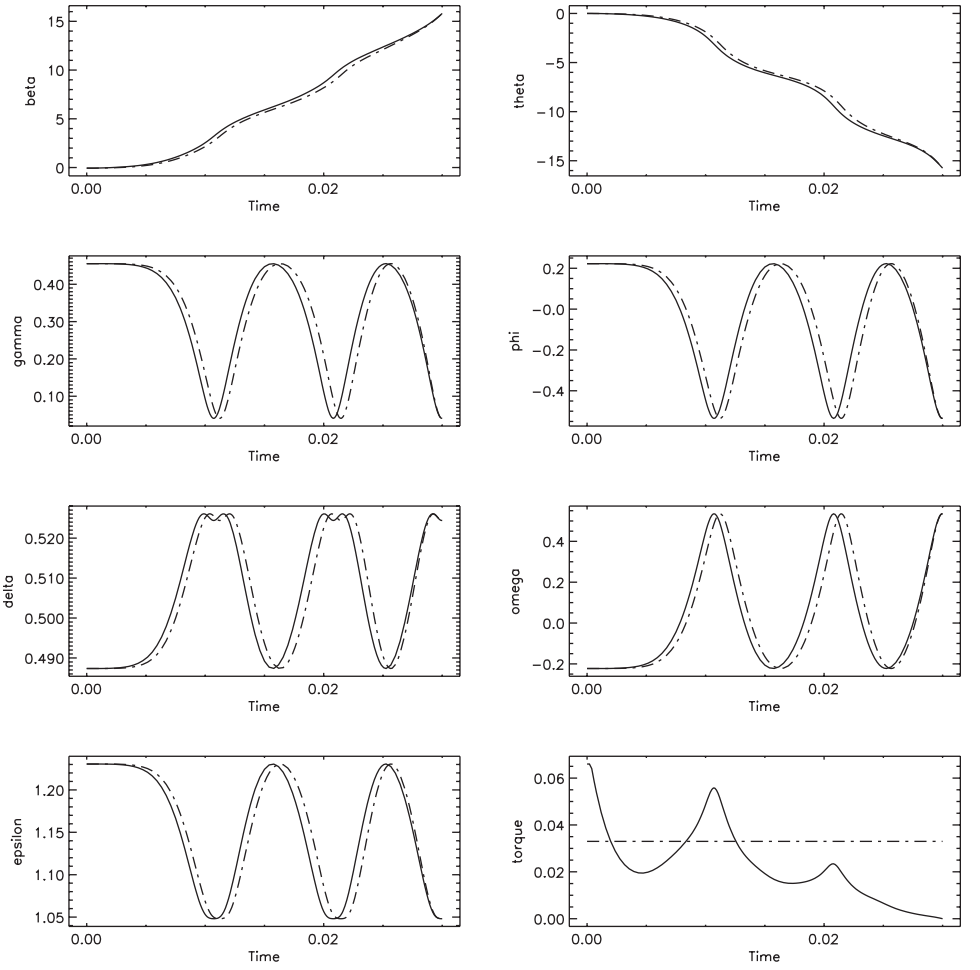


Figure 6.38. *Squeezer mechanism solution.*

one obtains the requisite explicit expression for $\mathbf{q} = \mathbf{q}(\mathbf{v}, \mathbf{p}, u_0)$. Of course, in practice we just solve (6.161) and do not explicitly compute the matrix inverse. An Adams predictor-corrector method was used to integrate the resulting ODE system, although any other IVP method could be used.

Figure 6.38 illustrates the solution obtained using $\mathbb{SOCS}$. The minimum energy results are plotted with a solid line and the constant-torque reference trajectories using a dashed line. The minimum energy $J^* = 0.6669897295$ compared with a value $J = 1$ for the constant-torque reference trajectory. The initial guess was constructed by evaluating the integrated profile at 20 equally spaced grid points. The final solution was obtained after four mesh-refinement iterations; the algorithm performance is summarized in Table 6.19. It is interesting to note that the preferred discretization selected by the mesh-refinement algorithm, as described in Section 4.6.9, was the HSS (HS). To more fully appreciate this behavior, the same problem was solved four different ways—using two different discretiza-

Table 6.19. *Minimum energy squeezer mechanism.*

Iter.	Disc.	M	GE	HE	FE	RHS	ϵ_{max}	CPU (sec)
1	HS	20	29	15	2585	100815	0.66×10^{-4}	0.15×10^2
2	HS	39	18	15	2381	183337	0.98×10^{-5}	0.22×10^2
3	HS	72	10	7	1154	165022	0.27×10^{-5}	0.18×10^2
4	HS	143	7	2	456	129960	0.24×10^{-7}	0.20×10^2
Total		143	64	39	6576	579134		74.15

tions, with and without right-hand-side sparsity. For the sake of discussion, we refer to these as Methods A–D and they are summarized below:

Method	Discretization	Separability	RHS sparsity
A	HS	Yes	Yes
B	HS	Yes	No
C	HC	No	Yes
D	HC	No	No

An examination of the data in Table 6.20 explains why Method A (HSS exploiting right-hand-side sparsity) is the preferred approach. First, the number of index sets was 16. This can be attributed both to the separability of the discretization and the right-hand-side sparsity as illustrated by the template (cf. (4.115))

$$\mathcal{T} = \operatorname{struct}\left[\begin{array}{c|c} \frac{\partial \mathbf{f}}{\partial \mathbf{y}} & \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \\ \frac{\partial \mathbf{g}}{\partial \mathbf{y}} & \frac{\partial \mathbf{g}}{\partial \mathbf{u}} \\ \frac{\partial \mathbf{w}}{\partial \mathbf{y}} & \frac{\partial \mathbf{w}}{\partial \mathbf{u}} \end{array}\right] = \left[\begin{array}{cccccccccccccccccccccccccccc}
&&&&&&&\times&&&&&&&&&&&&&&&&&&&& \\
&&&&&&&&\times&&&&&&&&&&&&&&&&&&& \\
&&&&&&&&&\times&&&&&&&&&&&&&&&&&& \\
&&&&&&&&&&\times&&&&&&&&&&&&&&&&& \\
&&&&&&&&&&&\times&&&&&&&&&&&&&&&& \\
&&&&&&&&&&&&\times&&&&&&&&&&&&&&& \\
&&&&&&&&&&&&&\times&&&&&&&&&&&&&& \\
&&&&&&&&&&&&&&\times&&&&&&&&&&&&& \\
&&&&&&&&&&&&&&&\times&&&&&&&&&&&& \\
&&&&&&&&&&&&&&&&\times&&&&&&&&&&& \\
&&&&&&&&&&&&&&&&&\times&&&&&&&&&& \\
&&&&&&&&&&&&&&&&&&\times&&&&&&&&& \\
&&&&&&&&&&&&&&&&&&&\times&&&&&&&& \\
&&&&&&&&&&&&&&&&&&&&\times&&&&&&& \\
\times&\times&&&&&&\times&\times&&&&&&\times&\times&&&&&&\times&\times&\times&\times&\times&\times&\times \\
\times&\times&&&&&&\times&&&&&&&\times&\times&&&&&&\times&\times&\times&\times&\times&\times& \\
&&\times&&&&&&&&&&&&&&\times&&&&&\times&\times&&&&& \\
&&&\times&\times&&&&&&&\times&&&&&&\times&\times&&&&&\times&\times&&& \\
&&&\times&\times&&&&&&\times&\times&&&&&&\times&\times&&&&&\times&\times&&& \\
&&&&&\times&\times&&&&&&&\times&&&&&&\times&\times&&&&&\times&\times& \\
&&&&&\times&\times&&&&&&\times&\times&&&&&&\times&\times&&&&&\times&\times& \\
\times&\times&\times&&&&&\times&\times&\times&&&&&\times&\times&\times&&&&&&&&&&& \\
\times&\times&\times&&&&&\times&\times&\times&&&&&\times&\times&\times&&&&&&&&&&& \\
\times&\times&&\times&\times&&&\times&\times&&\times&\times&&&\times&\times&&\times&\times&&&&&&&&& \\
\times&\times&&\times&\times&&&\times&\times&&\times&\times&&&\times&\times&&\times&\times&&&&&&&&& \\
\times&\times&&&&\times&\times&\times&\times&&&&\times&\times&\times&\times&&&&\times&\times&&&&&&& \\
\times&\times&&&&\times&\times&\times&\times&&&&\times&\times&\times&\times&&&&\times&\times&&&&&&& \\
&&&&&&&&&&&&&&&&&&&&&&&&&&&\times
\end{array}\right]. \tag{6.164}$$

When right-hand-side sparsity is not exploited (Method B), the number of index sets increases to 28. Furthermore, when discretization separability is not exploited, the number of index sets becomes larger still. Because the number of index sets for the HSS method is much smaller, the number of function evaluations needed to compute the Jacobian and Hessian is also significantly less. The net result is that the right-hand sides of the DAEs were

Table 6.20. *Discretization performance comparison.*

	A	B	C	D
Index sets	16	28	38	70
FE per Hess./Jac.	152	434	779	2555
Total RHS eval.	579134	1628010	2879084	8399250
Largest NLP	7980	9044	5992	6790
Mesh-ref. iter.	4	5	4	5
Total CPU time	74.15	179.84	151.43	400.37
% Time for FE	23%	34%	55%	70%

evaluated a significantly smaller number of times. Notice that the final NLP was larger for the HSS discretization (Methods A and B). Nevertheless, Method A is over 5.4 times faster than the HSC discretization without right-hand-side sparsity (Method D). In simple terms, for this problem it is better to solve a larger NLP problem because the derivatives can be computed more efficiently!

The solution to this problem exhibits another phenomenon associated with the numerical solution of high-index DAEs. Figure 6.39 plots the time history of the errors in the path constraints. Specifically, it displays the acceleration-level error $\|\mathbf{g}_{pp}(\mathbf{p})(\mathbf{v},\mathbf{v})+\mathbf{G}(\mathbf{p})\mathbf{q}\|$, the velocity-level error $\|\mathbf{G}(\mathbf{p})\mathbf{v}\|$, and the position error $\|\mathbf{g}(\mathbf{p})\|$. The acceleration error is acceptably small: $\|\frac{d^2\mathbf{g}(t)}{dt^2}\| \sim \mathcal{O}(10^{-11})$. However, both the position and the velocity errors "drift" significantly from zero.

6.11 Kinematic Chain

Example 6.15 KINEMATIC CHAIN. Büskens and Gerdts (see [92]) present an example that requires control of a *multibody system*. The problem is interesting because it can be made arbitrarily large and requires the treatment of an index-2 DAE system. The multibody dynamics are described by a system similar to (6.147)–(6.148):

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{6.165}$$

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{f}(\mathbf{p},\mathbf{v}) - \mathbf{C}^{\mathsf{T}}(\mathbf{p})\boldsymbol{\lambda} + \mathbf{K}\mathbf{u}, \tag{6.166}$$

$$\mathbf{0} = \mathbf{c}(\mathbf{p}). \tag{6.167}$$

Figure 6.40 illustrates a chain with ν links, which can be described by the position vector

$$\mathbf{p}^{\mathsf{T}} = \left(\mathbf{p}_1,\ldots,\mathbf{p}_{\nu+1}\right)^{\mathsf{T}}. \tag{6.168}$$

The location of the individual joints can be represented by four Cartesian components

$$\mathbf{p}_k = \begin{pmatrix}\mathbf{x}_k \\ \mathbf{d}_k\end{pmatrix}, \qquad k = 1,\ldots,\nu, \tag{6.169}$$

$$\mathbf{p}_{\nu+1} = \mathbf{x}_{\nu+1}, \tag{6.170}$$

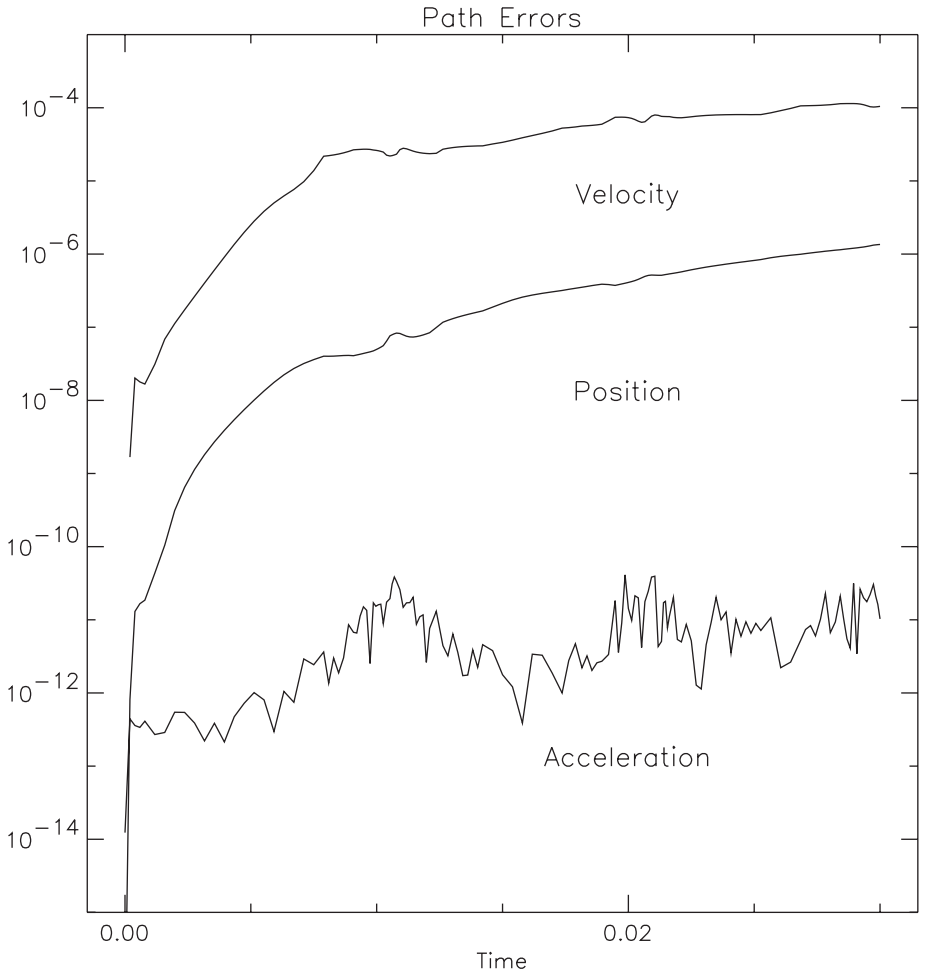


Figure 6.39. *Path-constraint errors.*

where the 2-vector $\mathbf{x}_k = (x_k, y_k)$ defines the horizontal and vertical position of link k, and the 2-vector $\mathbf{d}_k$ points from joint k to joint $(k+1)$. The *holonomic constraints* (6.167)

$$\mathbf{c}(\mathbf{p}) = \begin{bmatrix} \mathbf{c}_1(\mathbf{p}_1, \mathbf{x}_2) \\ \vdots \\ \mathbf{c}_\nu(\mathbf{p}_\nu, \mathbf{x}_{\nu+1}) \end{bmatrix} \tag{6.171}$$

restrict the motion of the individual links. Specifically links of length l_k must satisfy the conditions

$$\mathbf{c}_k(\mathbf{p}_k, \mathbf{x}_{k+1}) = \begin{bmatrix} \frac{1}{2}\left(\|\mathbf{d}_k\|^2 - l_k^2\right) \\ \mathbf{x}_k + \mathbf{d}_k - \mathbf{x}_{k+1} \end{bmatrix} = \mathbf{0}. \tag{6.172}$$

As written, (6.165)–(6.167) is an index-3 DAE system, and as such it is desirable to reduce

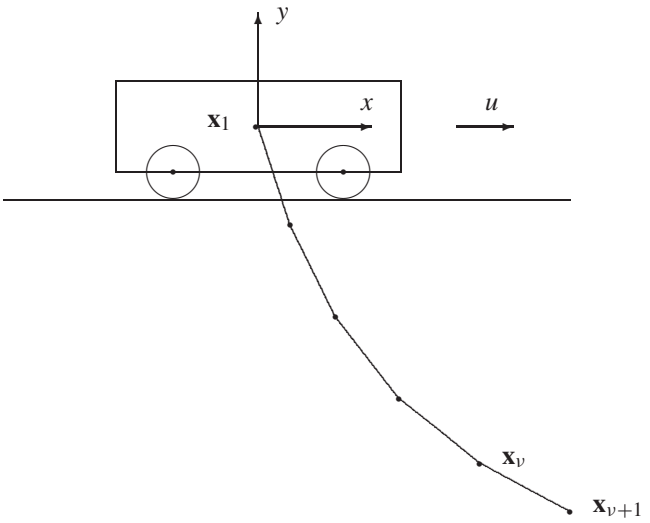


Figure 6.40. *Kinematic chain.*

the index by differentiating (6.167). In particular we must have

$$\mathbf{0} = \frac{d}{dt}\big[\mathbf{c}(\mathbf{p})\big] = \mathbf{C}(\mathbf{p})\dot{\mathbf{p}} = \mathbf{C}(\mathbf{p})\mathbf{v}. \tag{6.173}$$

The velocity constraint is represented in terms of the matrix

$$\mathbf{C}(\mathbf{p}) = \begin{bmatrix} \mathbf{C}_1(\mathbf{p}_1) & \mathbf{P}_1 & & \\ & \ddots & \ddots & \\ & & \mathbf{C}_\nu(\mathbf{p}_\nu) & \mathbf{P}_\nu \end{bmatrix}, \tag{6.174}$$

where

$$\mathbf{C}_k(\mathbf{p}_k) = \begin{bmatrix} (0,0) & \mathbf{d}_k^{\mathsf{T}} \\ \mathbf{I}_2 & \mathbf{I}_2 \end{bmatrix} \tag{6.175}$$

with

$$\mathbf{P}_k = \begin{bmatrix} \mathbf{0}_2 & \mathbf{0}_2 \\ -\mathbf{I}_2 & \mathbf{0}_2 \end{bmatrix}, \qquad k = 1, \ldots, (\nu - 1), \qquad\qquad \mathbf{P}_\nu = \begin{bmatrix} \mathbf{0}_2 \\ -\mathbf{I}_2 \end{bmatrix} \tag{6.176}$$

and

$$\mathbf{0}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad\qquad \mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

An index-one DAE can be obtained by again differentiating

$$\mathbf{0} = \frac{d}{dt}\big[\mathbf{C}(\mathbf{p})\mathbf{v}\big] = \dot{\mathbf{C}}(\mathbf{p})\mathbf{v} + \mathbf{C}(\mathbf{p})\dot{\mathbf{v}}. \tag{6.177}$$

From (6.174) it follows that

$$\dot{\mathbf{C}}(\mathbf{p}) = \begin{bmatrix} \dot{\mathbf{C}}_1(\mathbf{p}_1) & \dot{\mathbf{P}}_1 & & \\ & \ddots & \ddots & \\ & & \dot{\mathbf{C}}_\nu(\mathbf{p}_\nu) & \dot{\mathbf{P}}_\nu \end{bmatrix}, \tag{6.178}$$

where

$$\dot{\mathbf{C}}_k(\mathbf{p}_k) = \begin{bmatrix} (0,0) & \dot{\mathbf{d}}_k^\top \\ \mathbf{0}_2 & \mathbf{0}_2 \end{bmatrix} = \begin{bmatrix} (0,0) & (v_{k,3}, v_{k,4}) \\ \mathbf{0}_2 & \mathbf{0}_2 \end{bmatrix} \tag{6.179}$$

and $\dot{\mathbf{P}}_k = \mathbf{0}$.

Finally, by adding an additional algebraic variable $\mathbf{q}$ and path constraint, (6.166) can be written in semiexplicit form. Collecting results the dynamics are defined by the index-one DAE system

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{6.180}$$
$$\dot{\mathbf{v}} = \mathbf{q}, \tag{6.181}$$
$$\mathbf{0} = \mathbf{M}\mathbf{q} - \mathbf{f}(\mathbf{p}, \mathbf{v}, \mathbf{u}) + \mathbf{C}^\mathsf{T}(\mathbf{p})\boldsymbol{\lambda} - \mathbf{K}\mathbf{u}, \tag{6.182}$$
$$\mathbf{0} = \dot{\mathbf{C}}\mathbf{v} + \mathbf{C}\mathbf{q}. \tag{6.183}$$

The problem description is completed by defining the quantities in (6.182). First, the mass matrix is given by

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_1 & & & & \\ & \mathbf{M}_2 & & & \\ & & \ddots & & \\ & & & \mathbf{M}_\nu & \\ & & & & \mathbf{0}_2 \end{bmatrix} \tag{6.184}$$

with

$$\mathbf{M}_1 = \left(2 + \nu^{-1}\right) \begin{bmatrix} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{12} \end{bmatrix} \tag{6.185}$$

and

$$\mathbf{M}_k = \nu^{-1} \begin{bmatrix} \mathbf{I}_2 & \frac{1}{2}\mathbf{I}_2 \\ \frac{1}{2}\mathbf{I}_2 & \frac{1}{3}\mathbf{I}_2 \end{bmatrix}, \qquad k = 2, \ldots, \nu. \tag{6.186}$$

The force vector is given by

$$\mathbf{f}^\mathsf{T}(\mathbf{p}, \mathbf{v}) = (\mathbf{f}_1^\mathsf{T}, \mathbf{f}_2^\mathsf{T}, \ldots, \mathbf{f}_\nu^\mathsf{T}, 0, 0) \tag{6.187}$$

with

$$\mathbf{f}_1^\mathsf{T} = (0,0,0,0) \tag{6.188}$$

and

$$\mathbf{f}_k^{\mathsf{T}} = -g\nu^{-1}\left(0,1,0,\frac{1}{2}\right), \qquad k = 2,\ldots,\nu. \tag{6.189}$$

The constants correspond to a chain with a total mass of 1 kg, and links of length $l_k = \nu^{-1}$, for a total length of 1 m, where $g = 9.81$. There is a single control $\mathbf{u}(t) = u(t)$ corresponding to a force in the horizontal direction at link one. The matrix $\mathbf{K}$ is diagonal with

$$K_{i,i} = \begin{cases} 1, & i = 1, \\ 0, & i = 2,\ldots,4\nu+2. \end{cases} \tag{6.190}$$

The chain is initialized in a horizontal stretched out position

$$\mathbf{p}_k(t_I) = \begin{pmatrix} \mathbf{x}_k \\ \mathbf{d}_k \end{pmatrix} = \begin{pmatrix} (k-1)l_k \\ 0 \\ l_k \\ 0 \end{pmatrix}, \qquad k = 1,\ldots,\nu, \tag{6.191}$$

$$\mathbf{p}_{\nu+1}(t_I) = \mathbf{x}_{\nu+1} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{6.192}$$

with zero initial velocity $\mathbf{v}(t_I) = \mathbf{0}$. As an objective Büskens and Gerdts suggest minimizing

$$\begin{aligned} F &= \int_{t_I}^{t_F} \alpha \mathbf{x}_1^{\mathsf{T}}(t)\mathbf{x}_1(t) + \alpha^{-1}\mathbf{u}^{\mathsf{T}}(t)\mathbf{u}(t)dt \\ &= \alpha \int_{t_I}^{t_F} x_1^2(t)dt + \alpha \int_{t_I}^{t_F} y_1^2(t)dt + \frac{1}{\alpha}\int_{t_I}^{t_F} u^2(t)dt \end{aligned} \tag{6.193}$$

with $\alpha = 1000$ over the time interval $t_I = 0$ to $t_F = 1$.

To begin the optimization process a reasonable initial guess can be constructed by exploiting the algebraic equations (6.182) and (6.183) which can be written as

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^{\mathsf{T}} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f} + \mathbf{K}\mathbf{u} \\ -\dot{\mathbf{C}}\mathbf{v} \end{bmatrix}. \tag{6.194}$$

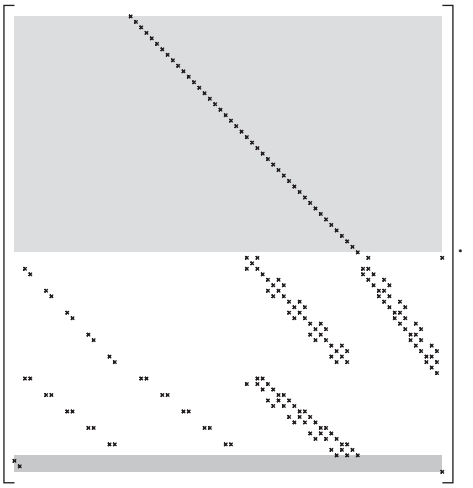
As an initial guess we choose $\mathbf{u}^{(0)}(t) = \mathbf{0}$. We then numerically integrate the ODE (not DAE) system (6.180)–(6.181) from the given initial conditions (6.191)–(6.192). To solve this IVP at each integration step it is necessary to compute $\mathbf{q}(t)$, which can be computed by solving the linear system (6.194). This procedure will construct an initial time history for all of the dynamic variables that satisfies the DAE system (6.180)–(6.183).

Computational results are presented for a chain with five links ($\nu = 5$), and Table 6.21 summarizes a number of key parameters for this case.

Table 6.21. *Chain problem parameters.*

Differential Variables	44
Algebraic Variables	38
Differential Equations	44
Algebraic Equations	37
Quadrature Functions	3
Max. Nonzeros in Row of RHS Template	6
Evaluations for Jacobian (HS)	12
Evaluations for Hessian (HS)	27
Evaluations for Jacobian (HC)	21
Evaluations for Hessian (HC)	42

For this example the right-hand-side sparsity template (cf. (4.89)) is of the form

$\mathcal{T} =$  .

The first 44 rows correspond to the differential equations (6.180)–(6.181) and are shaded a light gray. The next 37 rows correspond to the algebraic equations (6.182)–(6.183) and are unshaded. The final three rows, shaded a dark gray, correspond to the individual terms in the quadrature objective function (6.193). Observe that this integral has been written as the sum of three terms in order to exploit separability. It is critical to exploit the fact that the right-hand-side template has at most six nonzero elements in any row. In fact, if right-hand-side sparsity is ignored for this DAE system of order 82, a central difference Jacobian and Hessian will require 3485 function evaluations, which is significantly more than the values in Table 6.21. This point is further emphasized by reviewing the mesh-refinement summary in Table 6.22. Notice that the entire problem, including five mesh-refinement iterations, 60 gradient evaluations, and 29 Hessian evaluations, required only 4937 function evaluations. For this example the first refinement iteration utilized 25 equidistributed grid points and the HSS discretization. Subsequent iterations used an HSC discretization with the grid distribution constructed by the refinement algorithm.

Figure 6.41 shows the time history for all of the link position state variables, and the optimal control time history is illustrated in Figure 6.42. The optimal objective function is $F^* = .6447976339 \times 10^{-1}$.

Table 6.22. *Chain mesh-refinement summary.*

k	M	n	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	25	4018	35	13	586	37518	2.8163×10^{-3}	1.8730×10^{1}
2	49	5842	9	6	1626	188504	1.2506×10^{-4}	1.8970×10^{1}
3	97	11602	6	4	1077	265747	1.1385×10^{-5}	3.7610×10^{1}
4	193	23122	6	4	1077	523411	5.0244×10^{-7}	6.8140×10^{1}
5	385	46162	4	2	571	645353	3.2910×10^{-8}	9.1150×10^{1}
Total	385		60	29	4937	1660533		2.3460×10^{2}

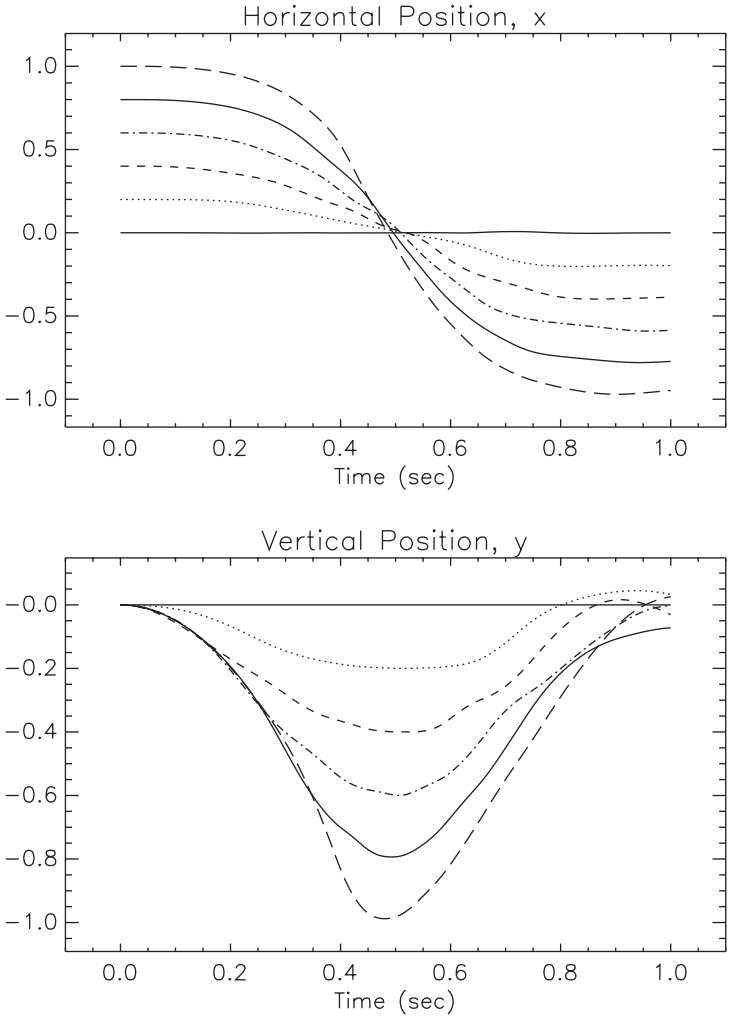


Figure 6.41. *Kinematic chain, states* $\mathbf{x}_k = (x_k, y_k)$ *for* $k = 1,\ldots,6$.

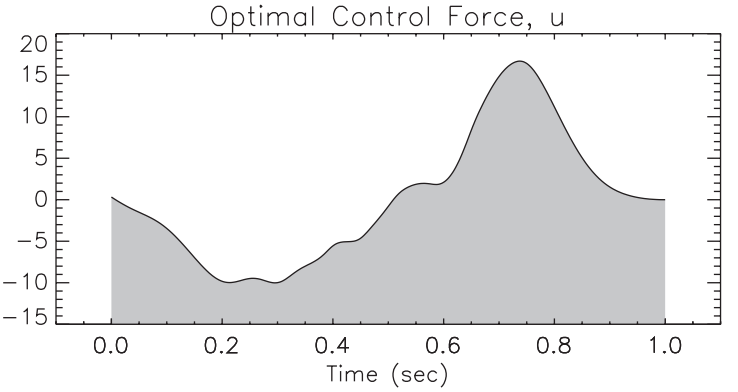


Figure 6.42. *Kinematic chain, optimal control.*

Solving this problem using the direct transcription method implemented in SOCS is relatively straightforward. In contrast, if a shooting or multiple shooting method is used, one must address a particular issue encountered for all DAE optimization problems. Specifically when the system dynamics are described by the DAEs

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \mathbf{u}, t), \tag{6.195}$$

$$\mathbf{0} = \mathbf{g}(\mathbf{y}, \mathbf{u}, t) \tag{6.196}$$

the shooting method must incorporate software for solving a DAE IVP such as DASSL [140, 141] or RADAU5 [107]. To start the integration it is necessary to have *consistent initial conditions*; that is, at the beginning of the propagation it is necessary that

$$\mathbf{0} = \mathbf{g}[\mathbf{y}(t_I), \mathbf{u}(t_I), t_I]. \tag{6.197}$$

However, when one or more of the values $\mathbf{y}(t_I)$, $\mathbf{u}(t_I)$, or t_I is modified as part of an optimization iteration, it is quite likely that at the perturbed point $(\tilde{\mathbf{y}}(t_I), \tilde{\mathbf{u}}(t_I), \tilde{t}_I) \neq (\mathbf{y}(t_I), \mathbf{u}(t_I), t_I)$

$$\mathbf{0} \neq \mathbf{g}[\tilde{\mathbf{y}}(t_I), \tilde{\mathbf{u}}(t_I), \tilde{t}_I]. \tag{6.198}$$

Unfortunately, when this happens the DAEs (6.195)–(6.196) cannot be propagated, and the process fails. Gerdts [92] discusses two different remedies for this dilemma. It is worth emphasizing that having inconsistent initial conditions is simply not an issue with the direct transcription method.

6.12 Dynamic MPEC

Example 6.16 DYNAMIC MPEC. When formulating the behavior of physical systems it is sometimes useful to describe the dynamics using differential equations with discontinuous right-hand sides. To illustrate this situation let us consider a simple example originally suggested by Stewart and Anitescu [162] and discussed by Baumrucker, Renfro,

and Biegler [7].[6] The goal is to minimize the objective

$$F = [y(2) - 5/3]^2 + \int_0^2 y^2(t)dt \tag{6.199}$$

and satisfy the simple differential equation

$$\dot{y} = 2 - sgn(y), \tag{6.200}$$

defined for $0 \le t \le 2$, with initial condition $y(0) = y_0$, where the signum or sign function is given by (1.126). If the state y represents the velocity of a body moving in contact with a surface, then the term $sgn(y)$ describes the friction force, and the example represents the situation found in contact and friction problems in computational mechanics. As stated there is no control variable and the solution is uniquely defined by the initial condition. However, as a practical matter the numerical solution of (6.200) is problematic because $sgn(y)$ is discontinuous.

Now, if $y(0) = y_0 = -1$, it is easy to demonstrate for this very simple example that the dynamics are described by

$$\dot{y} = 3, \qquad 0 \le t \le 1/3, \tag{6.201a}$$

$$\dot{y} = 1, \qquad 1/3 \le t \le 2. \tag{6.201b}$$

Clearly the best way to treat the discontinuity is to introduce a separate phase corresponding to each region. The location of the phase boundary can be determined by introducing a single variable t_s and then imposing a single continuity constraint $y(t_s^-) = y(t_s^+)$ between the phases. Unfortunately, for realistic problems with more complicated nonlinear dynamics it is seldom so easy to identify the phase structure. Instead the discontinuous behavior may occur many times during the process. Furthermore both the number and location of the discontinuities can change from one optimization iteration to the next.

Let us consider an alternate approach. Since $sgn[y(t)]$ is a discontinuous function of time, let us replace it by an algebraic variable

$$s(t) = sgn[y(t)], \tag{6.202}$$

which, unlike a state variable, can also behave discontinuously. Suppose at *every time* t we compute the variable $s(t) = s$ to minimize

$$\phi(s) = -sy \tag{6.203}$$

[6]This topic was the subject of a technical interchange at Argonne National Laboratory on October 25, 2004, hosted by Sven Leyffer, with participation by Larry Biegler, Shiva Kameswaran, and Juan J. Arrieta-Camacho (Carnegie Mellon University) and Steve Campbell (North Carolina State University).

subject to the constraints

$$s+1 \geq 0, \tag{6.204}$$
$$1-s \geq 0. \tag{6.205}$$

This is an inner level NLP just as first introduced in (1.128a)–(1.128b), and since it must be solved for each time t the overall approach is referred to as a *dynamic MPEC*.

Let us now formulate an optimal control problem to reflect these conditions. For this example we have a single differential variable $y(t)$ and three algebraic variables $\mathbf{u}^{\mathsf{T}}(t) = [s(t), p(t), q(t)]$. We must determine these quantities to minimize the objective function

$$F = [y(2) - 5/3]^2 + \int_0^2 y^2(t) + \rho\{p(t)[s(t)+1] + q(t)[1-s(t)]\}dt \tag{6.206}$$

for $\rho > 0$ and satisfy the dynamic constraints

$$\dot{y} = 2 - s(t), \tag{6.207}$$
$$0 = -y(t) - p(t) + q(t), \tag{6.208}$$
$$-1 \leq s(t) \quad \leq \quad 1, \tag{6.209}$$
$$0 \leq p(t), \tag{6.210}$$
$$0 \leq q(t) \tag{6.211}$$

with boundary condition $y(0) = y_0$. Observe that the complementarity conditions corresponding to (1.130)–(1.131)

$$0 \leq p \perp (s+1) \geq 0, \tag{6.212}$$
$$0 \leq q \perp (1-s) \geq 0 \tag{6.213}$$

are treated using an exact penalty function in the modified objective (6.206). When viewed this way, the functions $p(t)$ and $q(t)$ are just the time varying Lagrange multipliers for the inner level optimization problem (6.203)–(6.205). By using the integral form, the $\mathbb{SOCS}$ mesh-refinement procedure will automatically locate the grid points to achieve the required accuracy. The solution produced by $\mathbb{SOCS}$ is plotted in Figure 6.43. Table 6.23 summarizes the behavior of $\mathbb{SOCS}$ to solve this problem. The first two refinement iterations used a trapezoidal discretization and the last five were forced to be (compressed) Hermite–Simpson. The default $\mathbb{SOCS}$ approach would use (separated) Simpson for the last five refinements, and is a bit faster, but has the same grid configuration. Although this was done as a one-phase problem it appears that the switch occurs very close to $t = 1/3$ with almost all of the grid points clustered in this region as illustrated in Figure 6.44. For $x(0) = -1$

the optimal objective function value was $\phi^* = 1.6551964$, using the exact penalty value $\rho = 10^3$.

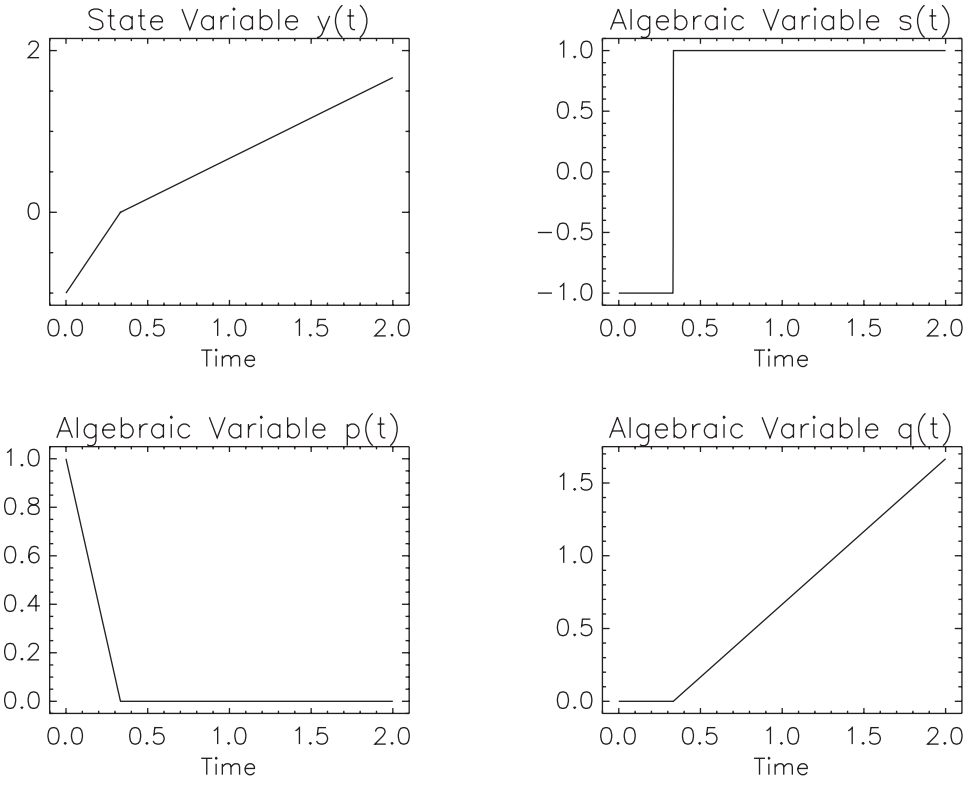


Figure 6.43. *MPEC solution.*

Table 6.23. *Mesh-refinement summary.*

k	M	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	10	6	3	45	2117	2.4691×10^{-1}	1.0×10^{-2}
2	16	11	9	101	4159	5.8964×10^{-3}	3.0×10^{-2}
3	16	8	6	503	21667	1.8081×10^{-3}	5.0×10^{-2}
4	24	8	6	503	32051	1.2141×10^{-4}	8.0×10^{-2}
5	30	41	39	3077	191545	6.9649×10^{-6}	6.1×10^{-1}
6	36	5	3	269	30749	1.7732×10^{-7}	1.8×10^{-1}
7	41	5	3	269	34819	$4.3035\times10^{-}8$	2.0×10^{-1}
Total	41	84	69	4767	317107		1.16

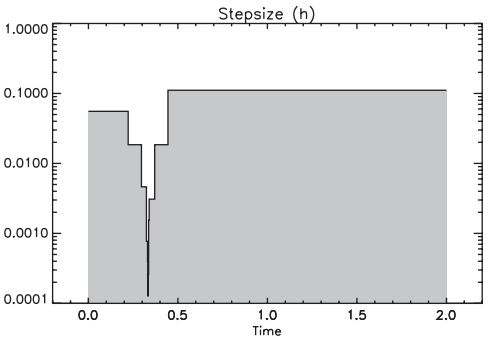


Figure 6.44. *MPEC grid distribution.*

6.13 Free-Flying Robot

Example 6.17 FREE-FLYING ROBOT. Sakawa [153] presents an example that describes the motion of a free-flying robot equipped with a propulsion system. The inertial coordinates of the center of gravity are denoted by (x_1, x_2) and the corresponding velocity by (v_1, v_2). The thrust direction is denoted by θ and the angular velocity by ω. The thrust from two engines denoted by T_1 and T_2 serve as the control variables:

$$\dot{x}_1 = v_1, \tag{6.214}$$
$$\dot{x}_2 = v_2, \tag{6.215}$$
$$\dot{\theta} = \omega, \tag{6.216}$$
$$\dot{v}_1 = [T_1 + T_2]\cos\theta, \tag{6.217}$$
$$\dot{v}_2 = [T_1 + T_2]\sin\theta, \tag{6.218}$$
$$\dot{\omega} = \alpha T_1 - \beta T_2. \tag{6.219}$$

Bounds are also imposed on the thrust magnitudes:

$$|T_1(t)| \le 1, \qquad |T_2(t)| \le 1. \tag{6.220}$$

As boundary conditions for the state $\mathbf{y} = (x_1, x_2, \theta, v_1, v_2, \omega)^{\mathsf{T}}$ we require

$$y(0) = \left(-10, -10, \frac{\pi}{2}, 0, 0, 0\right)^{\mathsf{T}}, \tag{6.221}$$
$$y(t_F) = (0,0,0,0,0,0)^{\mathsf{T}}. \tag{6.222}$$

Although the dynamics are relatively straightforward this problem is illustrative because of the objective function. The objective proposed by Sakawa was

$$F = \frac{1}{2}\sigma \|\mathbf{y}(t_F) - \mathbf{y}_F\|^2 + \gamma \int_0^{t_F} (|T_1| + |T_2|)\, dt. \tag{6.223}$$

The first term can be omitted by simply imposing the final condition (6.222) directly. Set-

ting $\gamma = 1$, $\alpha = \beta = .2$, and $t_F = 12$ completes the problem definition, and the objective function becomes

$$F = \int_0^{12} (|T_1| + |T_2|)\, dt. \tag{6.224}$$

Unfortunately, the objective function as written has discontinuous derivatives because it involves the absolute value function. An approach for treating absolute values motivated by the MPEC formulation (1.133c)–(1.133e) was described in Example 1.13, and the same technique can be utilized here. The "trick" is to express each control in terms of its positive and negative component:

$$T_1 = u_1 - u_2, \tag{6.225}$$
$$T_2 = u_3 - u_4 \tag{6.226}$$

with the restriction that $\mathbf{u} = (u_1, u_2, u_3, u_4)^{\mathrm{T}} \geq \mathbf{0}$. In effect, each "real" control is split into two controls, but in so doing one can write

$$|T_1| = u_1 + u_2, \tag{6.227}$$
$$|T_2| = u_3 + u_4. \tag{6.228}$$

After making this transformation the modified formulation requires minimizing the differentiable objective function

$$F = \int_0^{12} (u_1 + u_2 + u_3 + u_4)\, dt \tag{6.229}$$

while solving the DAE system

$$\dot{y}_1 = y_4, \tag{6.230}$$
$$\dot{y}_2 = y_5, \tag{6.231}$$
$$\dot{y}_3 = y_6, \tag{6.232}$$
$$\dot{y}_4 = [u_1 - u_2 + u_3 - u_4] \cos y_3, \tag{6.233}$$
$$\dot{y}_5 = [u_1 - u_2 + u_3 - u_4] \sin y_3, \tag{6.234}$$
$$\dot{y}_6 = \alpha(u_1 - u_2) - \beta(u_3 - u_4), \tag{6.235}$$
$$1 \geq u_1 + u_2, \tag{6.236}$$
$$1 \geq u_3 + u_4 \tag{6.237}$$

with $u_k \geq 0$ for $k = 1, 2, 3, 4$.

The optimal objective function is $F^* = 7.910154646$. The optimal state histories are illustrated in Figure 6.45. The time histories for the controls $\mathbf{u}$ are plotted in Figure 6.46 as well as the time history for the "real" controls T_1 and T_2. It is not surprising that the control history has a "bang-bang" appearance since the controls appear linearly in the problem. Note also that the final mesh distribution displayed in Figure 6.47 has very small steps in the vicinity of the step discontinuities, which is necessary to achieve the requested accuracy. This step contraction requires 12 iterations in the mesh-refinement procedure as summarized in Table 6.24.

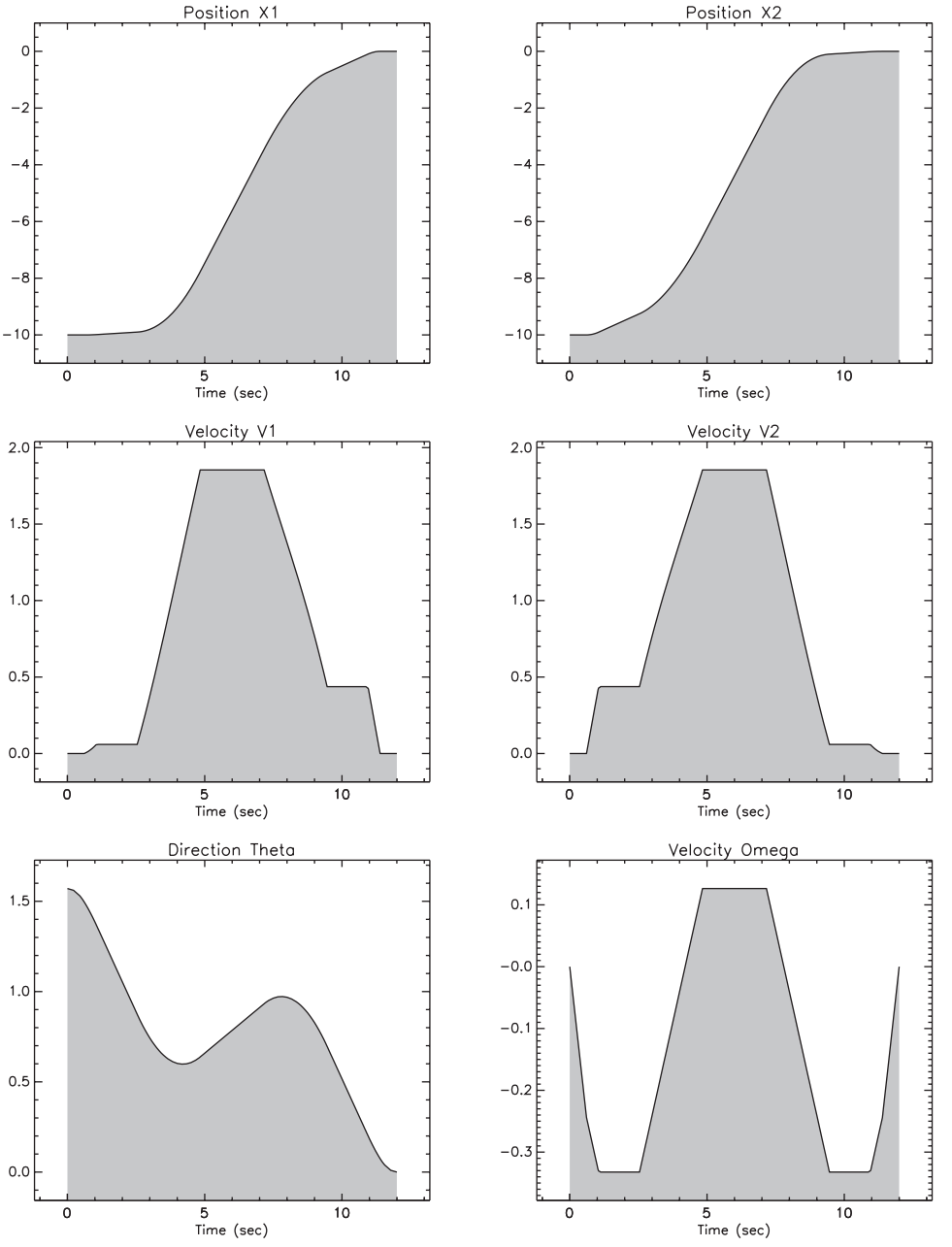


Figure 6.45. *Free-flying robot states.*

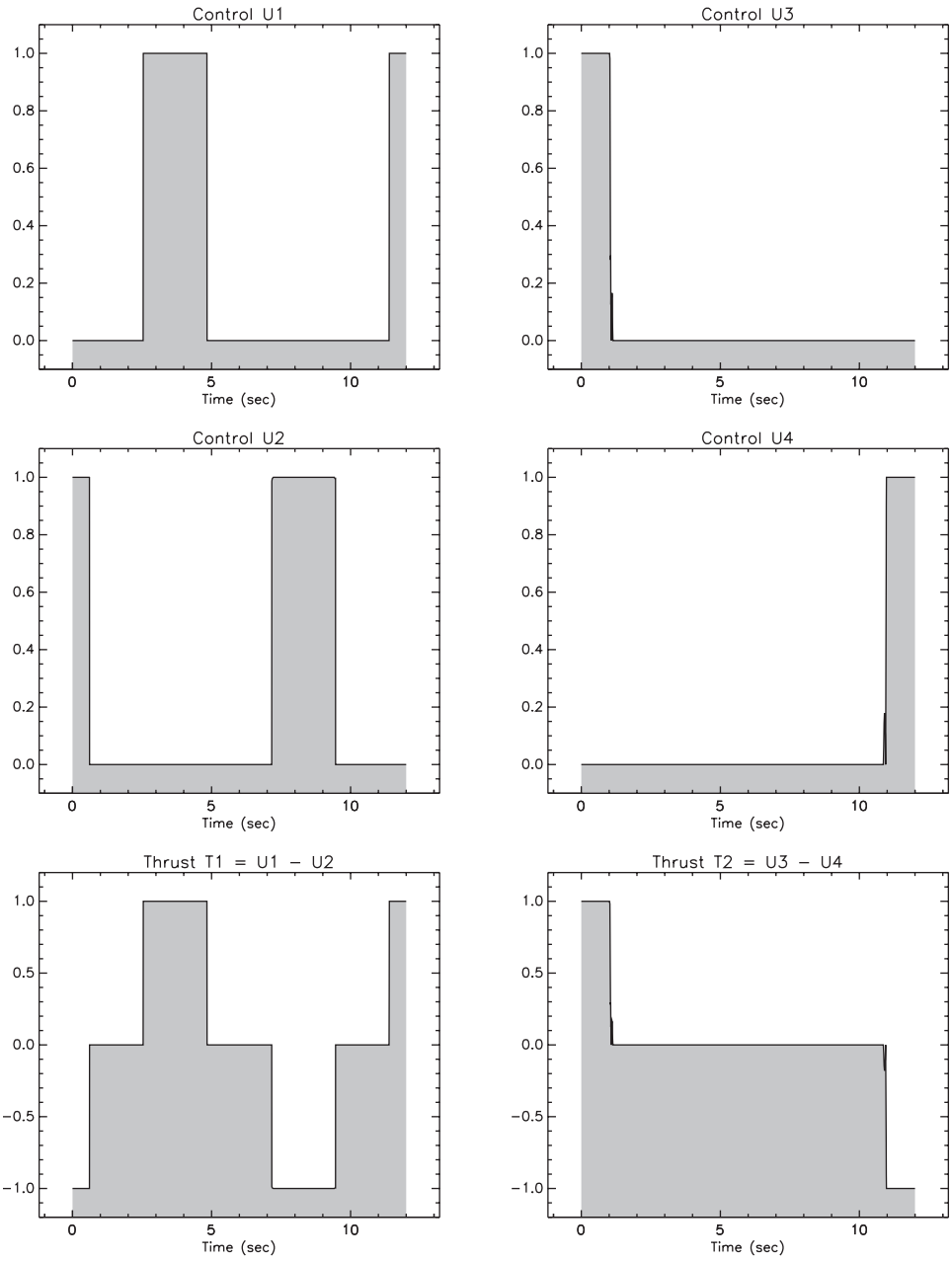


Figure 6.46. *Free-flying robot controls.*

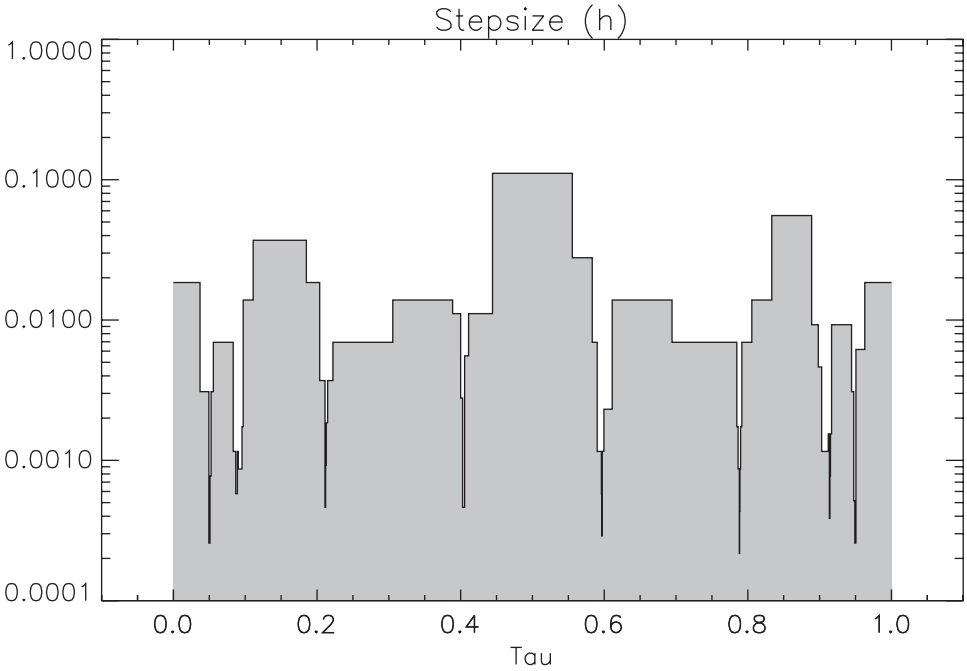


Figure 6.47. *Free-flying robot final grid distribution.*

Table 6.24. *Free-flying robot mesh-refinement summary.*

k	M	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	10	16	9	236	4417	3.8980×10^{-2}	9.0000×10^{-2}
2	10	10	8	1274	28968	1.4679×10^{-2}	1.2000×10^{-1}
3	19	11	9	1427	62133	3.5083×10^{-3}	3.8000×10^{-1}
4	37	28	26	4028	312522	3.6559×10^{-4}	$2.4200\times10^{+0}$
5	53	40	38	5864	641114	3.3852×10^{-4}	$5.2700\times10^{+0}$
6	69	20	18	2804	416454	1.8255×10^{-5}	$3.2700\times10^{+0}$
7	82	15	13	2039	370251	3.6540×10^{-6}	$6.0300\times10^{+0}$
8	103	44	42	6476	1375062	2.8825×10^{-5}	$1.6500\times10^{+1}$
9	115	17	15	2345	589151	4.4139×10^{-6}	$9.1200\times10^{+0}$
10	134	20	18	2804	808226	1.6770×10^{-6}	$1.2330\times10^{+1}$
11	142	7	5	815	293347	3.1444×10^{-7}	$9.7000\times10^{+0}$
12	159	5	3	509	230675	9.8315×10^{-8}	$1.1620\times10^{+1}$
Total	159	233	204	30621	5132320		$7.6850\times10^{+1}$

6.14 Kinetic Batch Reactor

Example 6.18 KINETIC BATCH REACTOR. In his doctoral thesis Leineweber [130] presents a problem originally given by Caracotsios and Stewart [62] that describes

> an optimal control problem which has several interesting features: stiff nonlinear DAE's, two model stages, a nonlinear inequality path constraint, equality and inequality boundary conditions, and unspecified terminal time. The example in its original form was given by the Dow Chemical Company as a challenging test problem for parameter estimation software .... The desired product AB is formed in the reaction
>
> $$HA + 2BM \longrightarrow AM + MBMH.$$
>
> (For proprietary reasons, the true nature of the reacting species has been disguised.)

Leineweber presents a kinetic model of the batch reactor system in terms of both differential and algebraic states denoted by x_j and z_j, respectively. A detailed explanation of the chemical process is omitted; however, Table 6.25 summarizes the model presented in [130]. All of the species concentrations (enclosed in brackets [.]) are given in gmol per kg of the reaction mixture. For the formulation given here the differential and algebraic variables are denoted by y_j and u_j, respectively, which correspond to the original notation as shown in Table 6.25.

Table 6.25. *Batch reactor dynamic variables.*

		Description	
y_1	x_0	Differential State	$[HA]+[A^-]$
y_2	x_1	Differential State	$[BM]$
y_3	x_2	Differential State	$[HABM]+[ABM^-]$
y_4	x_3	Differential State	$[AB]$
y_5	x_4	Differential State	$[MBMH]+[MBM^-]$
y_6	x_5	Differential State	$[M^-]$
u_1	z_0	Algebraic State	$-\log([H^+])$
u_2	z_1	Algebraic State	$[A^-]$
u_3	z_2	Algebraic State	$[ABM^-]$
u_4	z_3	Algebraic State	$[MBM^-]$
u_5	T	Reaction Temperature	

The kinetic model is stated in terms of six differential mass balance equations

$$\dot{y}_1 = -k_2 y_2 u_2, \tag{6.238}$$
$$\dot{y}_2 = -k_1 y_2 y_6 + k_{-1} u_4 - k_2 y_2 u_2, \tag{6.239}$$
$$\dot{y}_3 = k_2 y_2 u_2 + k_3 y_4 y_6 - k_{-3} u_3, \tag{6.240}$$
$$\dot{y}_4 = -k_3 y_4 y_6 + k_{-3} u_3, \tag{6.241}$$
$$\dot{y}_5 = k_1 y_2 y_6 - k_{-1} u_4, \tag{6.242}$$
$$\dot{y}_6 = -k_1 y_2 y_6 + k_{-1} u_4 - k_3 y_4 y_6 + k_{-3} u_3, \tag{6.243}$$

an electroneutrality condition

$$0 = p - y_6 + 10^{-u_1} - u_2 - u_3 - u_4, \tag{6.244}$$

and three equilibrium conditions

$$0 = u_2 - K_2 y_1/(K_2 + 10^{-u_1}), \tag{6.245}$$
$$0 = u_3 - K_3 y_3/(K_3 + 10^{-u_1}), \tag{6.246}$$
$$0 = u_4 - K_1 y_5/(K_1 + 10^{-u_1}), \tag{6.247}$$

where

$$k_1 = \hat{k}_1 \exp(-\beta_1/u_5), \tag{6.248}$$
$$k_{-1} = \hat{k}_{-1} \exp(-\beta_{-1}/u_5), \tag{6.249}$$
$$k_2 = \hat{k}_2 \exp(-\beta_2/u_5), \tag{6.250}$$
$$k_3 = k_1, \tag{6.251}$$
$$k_{-3} = \frac{1}{2} k_{-1}. \tag{6.252}$$

The values for the parameters $\hat{k}_j$ (kg/gmol/hr), β_j (K), and K_j (gmol/kg) are

$$\hat{k}_1 = 1.3708 \times 10^{12}, \quad \beta_1 = 9.2984 \times 10^3, \quad K_1 = 2.575 \times 10^{-16},$$
$$\hat{k}_{-1} = 1.6215 \times 10^{20}, \quad \beta_{-1} = 1.3108 \times 10^4, \quad K_2 = 4.876 \times 10^{-14},$$
$$\hat{k}_2 = 5.2282 \times 10^{12}, \quad \beta_2 = 9.5999 \times 10^3, \quad K_3 = 1.7884 \times 10^{-16}.$$

The reaction temperature T (K) is treated as the control variable and is limited to

$$293.15 \le u_5(t) \le 393.15 \tag{6.253}$$

for the duration of the process $0 \le t \le t_F$. Leineweber introduces a piecewise linear approximation for $u_5(t)$. In contrast, we will treat $u_5(t)$ like all other algebraic variables, and consequently the representation will be modified during the mesh-refinement procedure. Presumably this is a more accurate treatment for the reaction temperature control. The initial catalyst concentration, which Leineweber denotes by $[Q^+]$, is treated as the design parameter p (gmol/kg) appearing in (6.244). This parameter is included as an optimization variable and is restricted by the bounds

$$0 \le p \le .0262. \tag{6.254}$$

At the initial time $t = 0$ the parameter is related to the corresponding differential state through the point constraint

$$\psi = y_6(0) - p = 0 \tag{6.255}$$

and the remaining states are fixed by the boundary conditions

$$y_1(0) = 1.5776, \tag{6.256}$$
$$y_2(0) = 8.32, \tag{6.257}$$
$$y_3(0) = y_4(0) = y_5(0) = 0. \tag{6.258}$$

At the free final time t_F, it is required that

$$y_4(t_F) \geq 1. \tag{6.259}$$

In order to restrict the rate of product formation during the initial 25% of the process time, Leineweber introduces the nonlinear inequality path constraint

$$y_4(t) \leq at^2 \qquad \text{for} \quad 0 \leq t \leq (t_F/4), \tag{6.260}$$

where $a = 2$ (gmol/kg/hr^2). And finally, the desired objective is to minimize the quantity

$$F = \gamma_1 t_F + \gamma_2 p, \tag{6.261}$$

where $\gamma_1 = 1$ and $\gamma_2 = 100$.

The solution of this problem requires at least two phases, which Leineweber refers to as "stages," because the path constraint (6.260) must be imposed during the first portion of the process. Since the DAE system is stiff it is also helpful to introduce an additional phase at the beginning of the process in order to effectively model the rapid transient behavior. The domain is subdivided as follows:

$$0 \leq t \leq t_\epsilon \qquad \text{Phase 1}, \tag{6.262}$$
$$t_\epsilon \leq t \leq (t_F/4) \qquad \text{Phase 2}, \tag{6.263}$$
$$(t_F/4) \leq t \leq t_F \qquad \text{Phase 3}, \tag{6.264}$$

where we choose $t_\epsilon = .01$. Unlike the phase boundary at $t_F/4$, which must be introduced in order to treat the path constraint (6.260), the first phase is introduced strictly for numerical reasons. This rather simple artifice serves two purposes. First, it decouples the nonlinear transient behavior at the beginning of the process during the early iterations in much the same way as a multiple shooting method does. Second, it affords a simple way to construct an initial guess with a sufficiently fine mesh in the transient region. A piecewise linear initial guess with 40 grid points in Phase 1 and 30 in Phases 2 and 3 was used for the computational results. The three-phase description is completed by ensuring continuity in the differential states and the control across the phase boundaries, i.e.,

$$y_k^{(j)} = y_k^{(j+1)}, \qquad j = 1,2, \quad k = 1,\ldots,6, \tag{6.265}$$
$$u_5^{(j)} = u_5^{(j+1)}, \qquad j = 1,2. \tag{6.266}$$

The algebraic states (u_1, u_2, u_3, u_4) are implicitly linked across the phase boundaries by satisfying the algebraic path constraints (6.244)–(6.247). The differential and algebraic states are illustrated in Figures 6.48 and 6.49. The optimal control temperature is plotted in Figure 6.50. The piecewise linear guess for all dynamic variables is also plotted as a dashed line in the figures. Figure 6.51 illustrates the transient behavior during Phase 1 for some of the dynamic quantities. Table 6.26 summarizes the mesh-refinement history for this example.

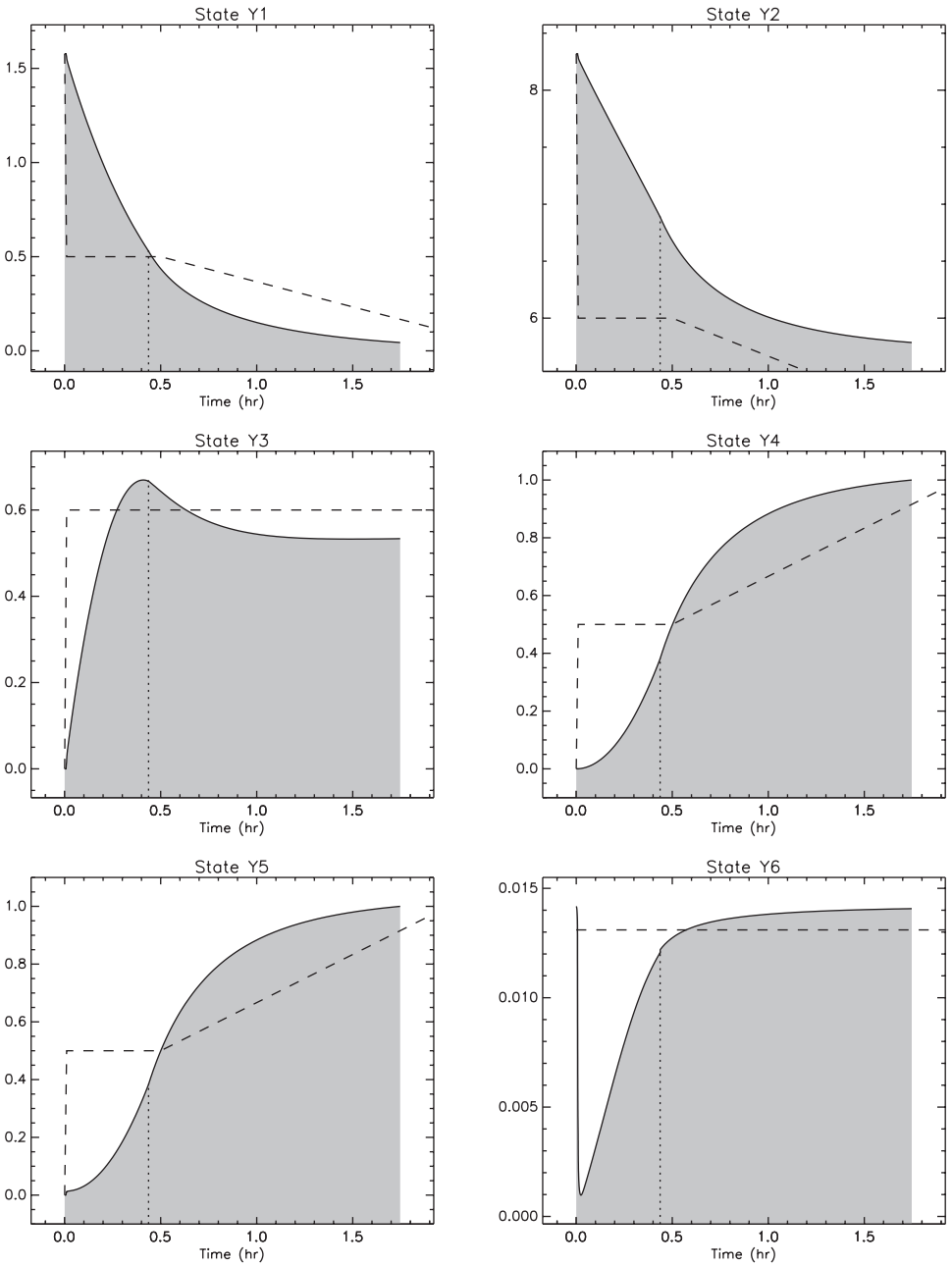


Figure 6.48. *Batch reactor differential states.*

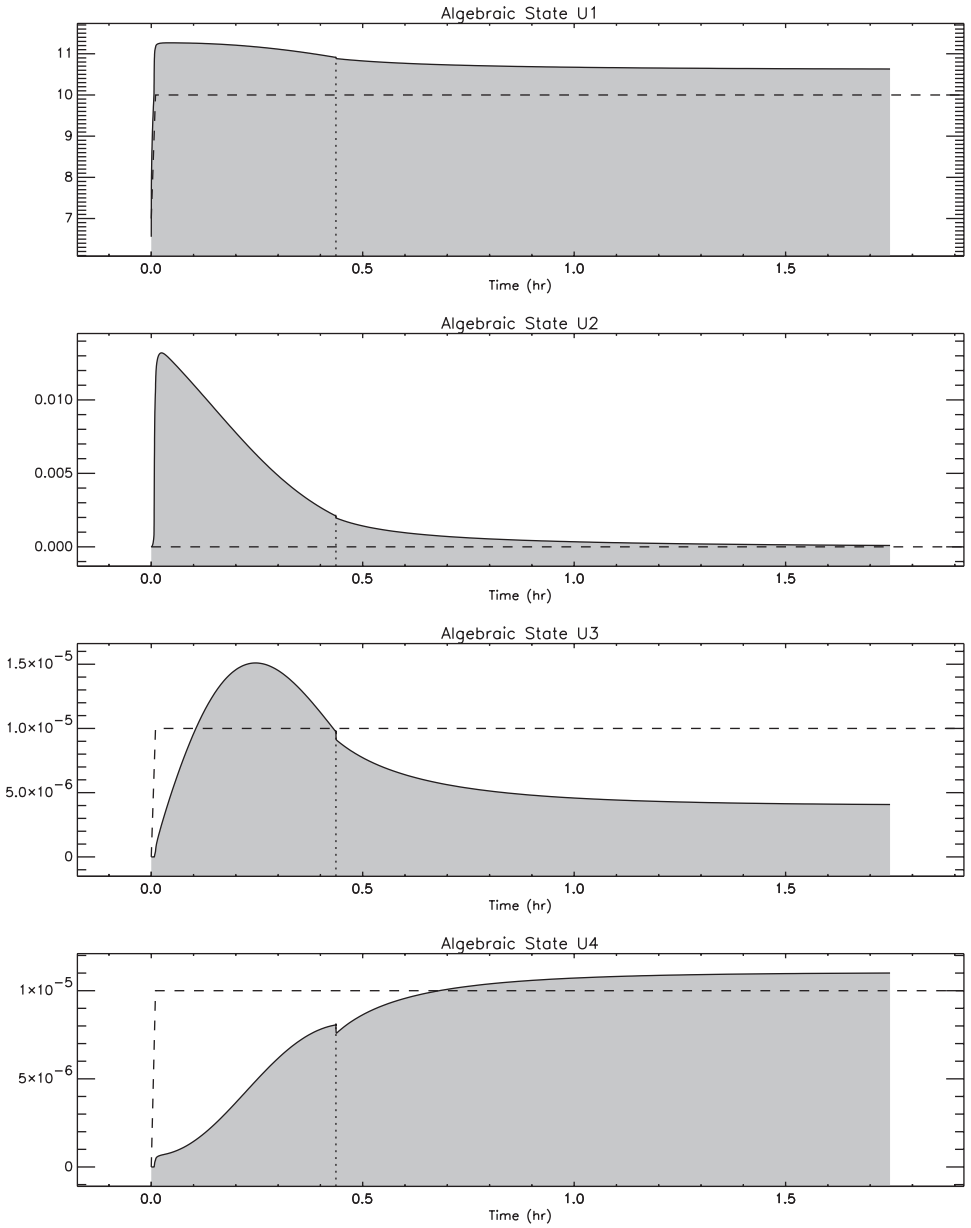


Figure 6.49. *Batch reactor algebraic states.*

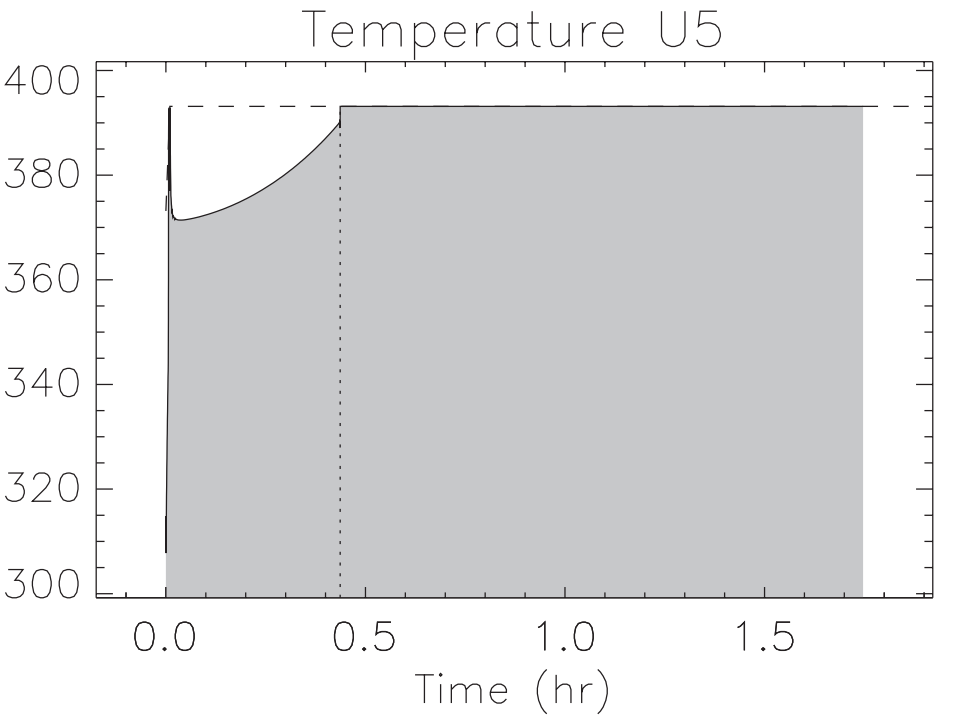


Figure 6.50. *Batch reactor control.*

6.15 Delta III Launch Vehicle

Example 6.19 DELTA III LAUNCH VEHICLE. In his doctoral thesis, Benson [8] presents a simplified formulation of an ascent trajectory for the Delta III launch vehicle. This problem is presented as an example for the software GPOCS [147]. The dynamic model for the motion of a nonlifting point mass in flight over a spherical rotating earth expressed in Cartesian (ECI) coordinates is given by the differential-algebraic system

$$\dot{\mathbf{r}} = \mathbf{v}, \tag{6.267}$$

$$\dot{\mathbf{v}} = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} + \frac{T}{m}\mathbf{u} + \frac{1}{m}\mathbf{D}, \tag{6.268}$$

$$\dot{m} = -\xi, \tag{6.269}$$

$$1 = \|\mathbf{u}\|, \tag{6.270}$$

$$R_E \le \|\mathbf{r}\|. \tag{6.271}$$

In this formulation $\mathbf{r}$ is the position vector, $\mathbf{v}$ is the velocity vector, T is the total vacuum thrust for all engines, and m is the total mass. The total mass flow rate of all engines is denoted by ξ. The complete state vector is $\mathbf{y}^{\mathsf{T}} = (\mathbf{r}^{\mathsf{T}}, \mathbf{v}^{\mathsf{T}}, m)$ and the control vector $\mathbf{u}$ defines the inertial thrust direction. The path constraint (6.270) guarantees that $\mathbf{u}$ is a unit vector and (6.271) ensures the altitude is positive. The drag force is given by

$$\mathbf{D} = -\frac{1}{2}C_D S\rho\|\mathbf{v}_r\|\mathbf{v}_r, \tag{6.272}$$

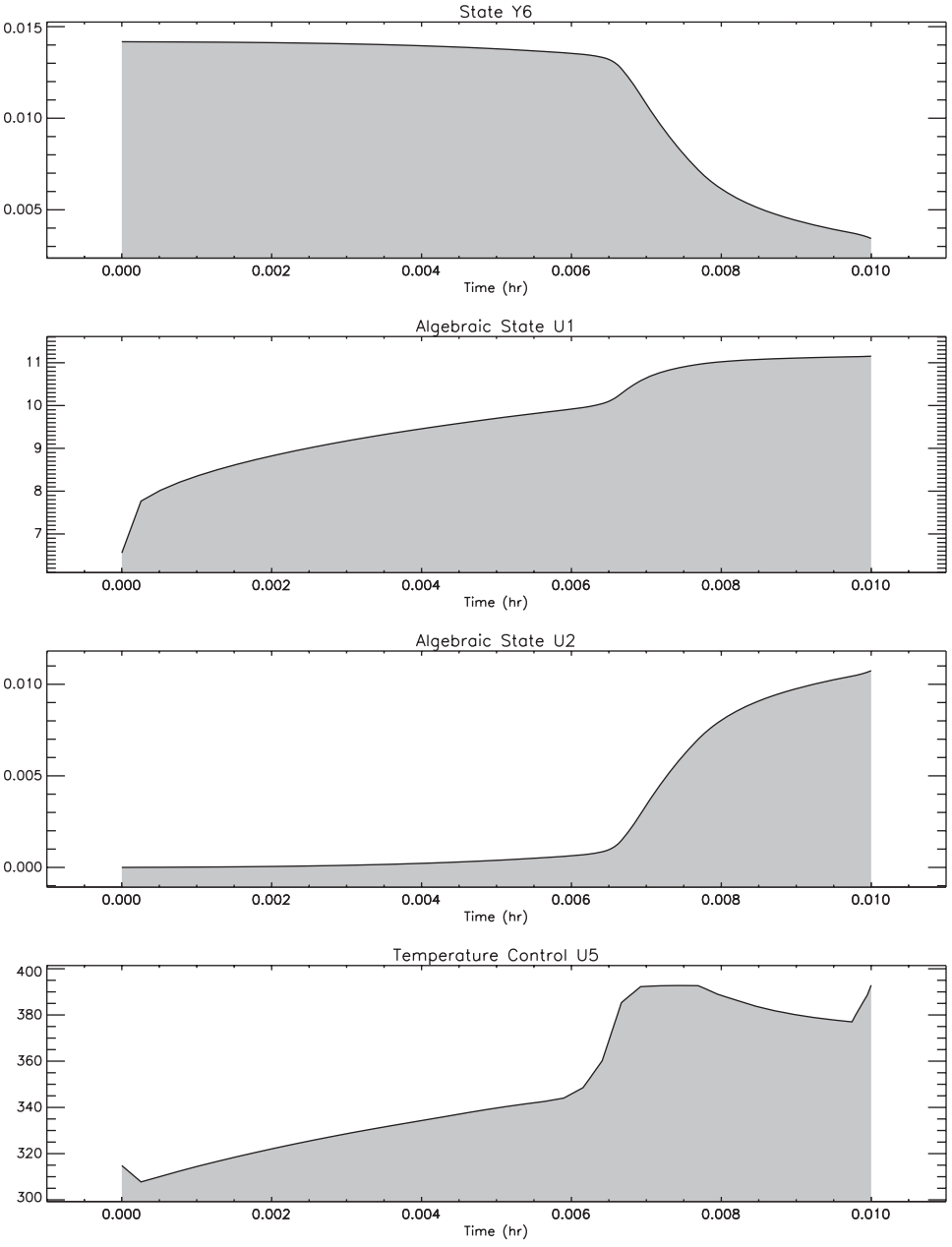


Figure 6.51. *Batch reactor transient behavior* ($0 \leq t \leq .01$).

Table 6.26. *Batch reactor mesh-refinement summary.*

k	M	NGC	NHC	NFE	NRHS	ϵ	Time (sec)
1	100	226	213	11859	1210606	1.4918×10^{-1}	$1.6920\times10^{+1}$
2	128	79	77	25099	3998429	4.5655×10^{-2}	$1.7170\times10^{+1}$
3	157	8	6	2024	736266	2.0728×10^{-2}	$7.5600\times10^{+0}$
4	236	7	5	1699	951477	1.5522×10^{-3}	$1.3580\times10^{+1}$
5	249	12	10	3324	1807486	1.8736×10^{-5}	$1.8260\times10^{+1}$
6	267	4	2	724	557582	3.1080×10^{-7}	$9.8100\times10^{+0}$
7	411	3	1	399	583047	5.0851×10^{-8}	$2.2290\times10^{+1}$
Total	411	339	314	45128	9844893		$1.0559\times10^{+2}$

where C_D is the drag coefficient, and S is the reference area. The aerodynamic force is defined in terms of the earth relative velocity vector

$$\mathbf{v}_r = \mathbf{v} - \boldsymbol{\omega} \times \mathbf{r}, \tag{6.273}$$

where $\boldsymbol{\omega}^{\mathsf{T}} = (0, 0, \omega_E)$ is the angular velocity of the earth relative to inertial space. The density ρ is modeled using a simple exponential atmosphere

$$\rho = \rho_0 e^{(-h/h_0)}, \tag{6.274}$$

where ρ_0 is the atmospheric density at sea level, $h = \|\mathbf{r}\| - R_E$ is the altitude, R_E is the spherical earth radius, and h_0 is the density scale height. Table 6.27 summarizes the parameters that define the dynamic model.

Table 6.27. *Dynamic model parameters.*

μ	3.986012×10^{14}	$\mathrm{m}^3/\mathrm{sec}^2$
R_E	6378145	m
g_0	9.80665	$\mathrm{m/sec}^2$
h_0	7200	m
ρ_0	1.225	$\mathrm{kg/m}^3$
ω_E	7.29211585×10^{-5}	rad/sec
C_D	.5	(nd)
S	4π	m^2

The Delta III expendable launch vehicle had two stages augmented with nine strap-on solid rocket boosters (SRBs). Because of the vehicle configuration it is natural to model the trajectory using four distinct phases. The first phase begins with the vehicle at rest on the ground, when the main engine and six of the nine solid rocket booster SRBs ignite. When the solid rocket burn to depletion, the inert weight is ejected. The second phase begins when the remaining three SRBs are ignited and terminates when the solid propellant is depleted. After ejecting the inert weight for the solid rockets, the third phase, consisting of the main engine only, continues until the remaining liquid propellant in stage 1 is depleted.

After ejecting the inert weight from stage 1, the second stage is ignited and burns until the final orbit is achieved, at the time of second engine cutoff (TSECO). Table 6.28 summarizes the parameters that define the vehicle.

Table 6.28. *Vehicle parameters.*

Description	Symbol	Solid Boosters	Stage 1	Stage 2
Total Mass, (kg)	ϖ	19290	104380	19300
Propellant Mass, (kg)	ϱ	17010	95550	16820
Structure Mass, (kg)	φ	2280	8830	2480
Engine Thrust, (N)	T	628500	1083100	110094
Specific Impulse, (sec)	$\mathcal{I}$	283.33364	301.68776	467.21311
Burn Time, (sec)	τ	75.2	261	700

The value for a particular component is denoted using the appropriate subscript; e.g., ϱ_s is the propellant mass of a solid booster. Note that unlike [8], the specific impulse is given by $\mathcal{I} = T\tau/(g_0\varrho)$, to ensure consistency with the other vehicle parameters.[7] The mass of the payload is $\varpi_p = 4164$ (kg).

Within each phase of the ascent trajectory the thrust T and mass flow ξ used in the dynamics (6.267)–(6.270) are defined as follows:

Phase 1 $0 = t_I^{(1)} \le t \le t_F^{(1)} = 75.2,$

$$T = 6T_s + T_1, \tag{6.275}$$

$$\xi = \frac{6T_s}{g_0\mathcal{I}_s} + \frac{T_1}{g_0\mathcal{I}_1}. \tag{6.276}$$

Phase 2 $75.2 = t_I^{(2)} \le t \le t_F^{(2)} = 150.4,$

$$T = 3T_s + T_1, \tag{6.277}$$

$$\xi = \frac{3T_s}{g_0\mathcal{I}_s} + \frac{T_1}{g_0\mathcal{I}_1}. \tag{6.278}$$

Phase 3 $150.4 = t_I^{(3)} \le t \le t_F^{(3)} = 261,$

$$T = T_1, \tag{6.279}$$

$$\xi = \frac{T_1}{g_0\mathcal{I}_1}. \tag{6.280}$$

Phase 4 $261 = t_I^{(4)} \le t \le t_F^{(4)} \le 961,$

$$T = T_2, \tag{6.281}$$

$$\xi = \frac{T_2}{g_0\mathcal{I}_2}. \tag{6.282}$$

[7] Computational results presented here use $g_0 = 9.80665$ m/sec^2 and $\omega_E = 7.29211585 \times 10^{-5}$rad/sec. The results in [147] use $g_0 = 9.79827953736866$ and $\omega_E = 7.272205216643040 \times 10^{-5}$.

Furthermore, this vehicle definition determines a mass time line as follows:

$$m[t_I^{(1)}] = 9\varpi_s + \varpi_1 + \varpi_2 + \varpi_p, \tag{6.283}$$

$$m[t_F^{(1)}] = m[t_I^{(1)}] - 6\varrho_s - \frac{\tau_s}{\tau_1}\varrho_1, \tag{6.284}$$

$$m[t_I^{(2)}] = m[t_F^{(1)}] - 6\varphi_s, \tag{6.285}$$

$$m[t_F^{(2)}] = m[t_I^{(2)}] - 3\varrho_s - \frac{\tau_s}{\tau_1}\varrho_1, \tag{6.286}$$

$$m[t_I^{(3)}] = m[t_F^{(2)}] - 3\varphi_s, \tag{6.287}$$

$$m[t_F^{(3)}] = m[t_I^{(3)}] - \left(1 - 2\frac{\tau_s}{\tau_1}\right)\varrho_1, \tag{6.288}$$

$$m[t_I^{(4)}] = m[t_F^{(3)}] - \varphi_1, \tag{6.289}$$

where $m[t_I^{(k)}]$ is the mass at the beginning of phase k, and $m[t_F^{(k)}]$ is the mass at the end of the phase.

The launch vehicle begins at rest relative to the earth with the following initial state vector:

$$\mathbf{r}(0) = \mathbf{r}_0 \;=\; [R_E \cos\psi_L \;,\; 0 \;,\; R_E \sin\psi_L]^{\mathsf{T}}, \tag{6.290}$$

$$\mathbf{v}(0) = \mathbf{v}_0 \;=\; -\boldsymbol{\omega} \times \mathbf{r}_0, \tag{6.291}$$

$$m(0) = m_0 \;=\; 9\varpi_s + \varpi_1 + \varpi_2 + \varpi_p, \tag{6.292}$$

where $\psi_L = 28.5$ (deg) is the (geocentric) latitude of the launch site at Cape Canaveral, and (6.291) ensures the relative velocity given by (6.273) is zero. At the final time $t_f = t_F^{(4)}$, the launch vehicle must insert the payload into a geosynchronous transfer orbit (GTO). The classical elements

$$a_f = 24361140\,(\text{m}), \tag{6.293}$$

$$e_f = .7308, \tag{6.294}$$

$$i_f = 28.5\,(\text{deg}), \tag{6.295}$$

$$\Omega_f = 269.8\,(\text{deg}), \tag{6.296}$$

$$\omega_f = 130.5\,(\text{deg}), \tag{6.297}$$

referred to as semimajor axis, eccentricity, inclination, right ascension of the ascending node (RAAN), and argument of perigee, respectively, can all be computed from the terminal state vector $(\mathbf{r}_f, \mathbf{v}_f) = (\mathbf{r}[t_F^{(4)}], \mathbf{v}[t_F^{(4)}])$. Continuity in the position and velocity from phase to phase is enforced by imposing the conditions

$$\mathbf{r}[t_F^{(k)}] = \mathbf{r}[t_I^{(k+1)}], \tag{6.298}$$

$$\mathbf{v}[t_F^{(k)}] = \mathbf{v}[t_I^{(k+1)}] \tag{6.299}$$

for $k = 1,2,3$. Finally, the mass is linked by imposing (6.285), (6.287), and (6.289), which incorporates the requisite jettison of engine structure.

The objective is to choose the control vector $\mathbf{u}(t)$, and final time t_f, to maximize the final mass

$$F = m(t_f) \tag{6.300}$$

subject to the DAE constraints (6.267)–(6.271), with initial conditions (6.290)–(6.292), boundary conditions (6.293)–(6.297), and linkage conditions (6.298), (6.299), (6.285), (6.287), and (6.289).

A guess is required to initiate the iterative solution of the optimal control problem, and for the results presented here a simple linear interpolant of the dynamic variables was utilized. Specifically a guess for the state is

$$\mathbf{y}(t) = \mathbf{y}_0 + (\mathbf{y}_f - \mathbf{y}_0)\frac{t}{t_f}, \tag{6.301}$$

where $0 \le t \le t_f$. The initial state $\mathbf{y}_0$ can be defined immediately from (6.290)–(6.292). If we guess $t_f = 961$, the final mass is just $m_f = m[t_F^{(4)}] = m[t_I^{(4)}] - \varrho_2$, where (6.289) is used. A guess for the final values of the position and velocity vector $(\mathbf{r}_f, \mathbf{v}_f)$ can be constructed from the values of the classical elements (6.293)–(6.297) provided a guess for the true anomaly ν_f is also supplied. One can simply guess $\nu_f = 0$ or guess a value of ν_f such that the altitude has a specified value. For the results given, we guess ν_f such that $h_f = 200$ (km). The initial guess for the control angles is just $\mathbf{u}^{\mathrm{T}}(t) = (1,0,0)$.

The optimal steering commands $\mathbf{u}(t)$ are plotted in Figure 6.52. Figure 6.53 illustrates the optimal solution for the position $\mathbf{r}(t)$ and velocity state $\mathbf{v}(t)$. Figure 6.54 displays the history for the mass $m(t)$, the altitude $h(t) = \|\mathbf{r}(t)\| - R_E$, and the velocity $\|\mathbf{v}(t)\|$. The optimal value for the final mass is $F^* = m^*(t_f) = 7529.712412$ (kg), which occurs when $t_f = 924.139$ (sec).

Table 6.29 summarizes some of the important performance characteristics of the $\mathbb{SOCS}$ algorithm. Four mesh-refinement iterations were required to reduce the discretization error below the requested tolerance $\epsilon_{\max} = 10^{-7}$. The first refinement iteration used a trapezoidal discretization (denoted "TR"), with 10 grid points in each phase. The resulting sparse NLP had $n = 401$ variables, with 76 degrees of freedom (NDOF). This nonlinear program required solving 91 QP subproblems, which was completed in 2.00 seconds of CPU time, and yields a discretization error of $\epsilon_{\max} = 7.4 \times 10^{-4}$. To improve the solution accuracy, the mesh-refinement procedure did two things: in phase 1 it changed the discretization technique from trapezoidal to separated Hermite–Simpson (HS), and in the remaining phases increased the number of grid points from 10 to 19. This larger NLP with 761 variables was solved in .16 seconds and required six QP subproblems. Two additional mesh-refinement iterations were required, all using a compressed Hermite–Simpson (HC) discretization to achieve the requested accuracy, with the overall solution process taking 3.13 sec of CPU time. Observe that the number of QP iterations needed to solve the NLP problems does not grow with the number of degrees of freedom!

In comparison, the Gauss pseudospectral method implemented in the GPOCS software was also used to solve this problem. Using 20 nodes per phase, with no mesh refinement, the NLP subproblem was solved using the SNOPT [94] NLP algorithm. This formulation leads to a problem with 864 variables and 694 constraints. The GPOCS/SNOPT algorithm requires 1674 major iterations (QP subproblems) compared with 64 for the $\mathbb{SOCS}$

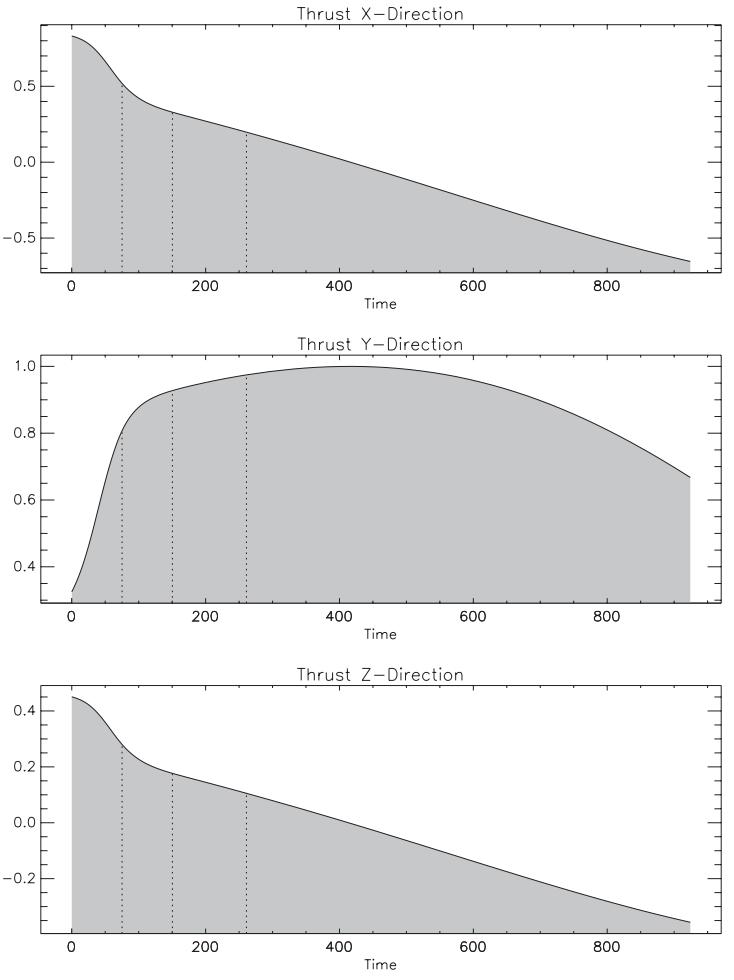


Figure 6.52. *Delta* III *ascent; controls* $\mathbf{u}(t)$.

results. The total solution time for the GPOCS/SNOPT algorithm was 2724.73 sec compared to 3.13 sec using $\mathbb{SOCS}$. The primary reason for this performance difference can be attributed to the underlying NLP algorithm. SNOPT constructs the Hessian matrix using a quasi-Newton method which requires 1674 iterations for this problem with 864 variables. The quasi-Newton approximation also does not exploit sparsity in the Hessian matrix. In contrast, $\mathbb{SOCS}$ is a Newton method and does exploit Hessian sparsity.

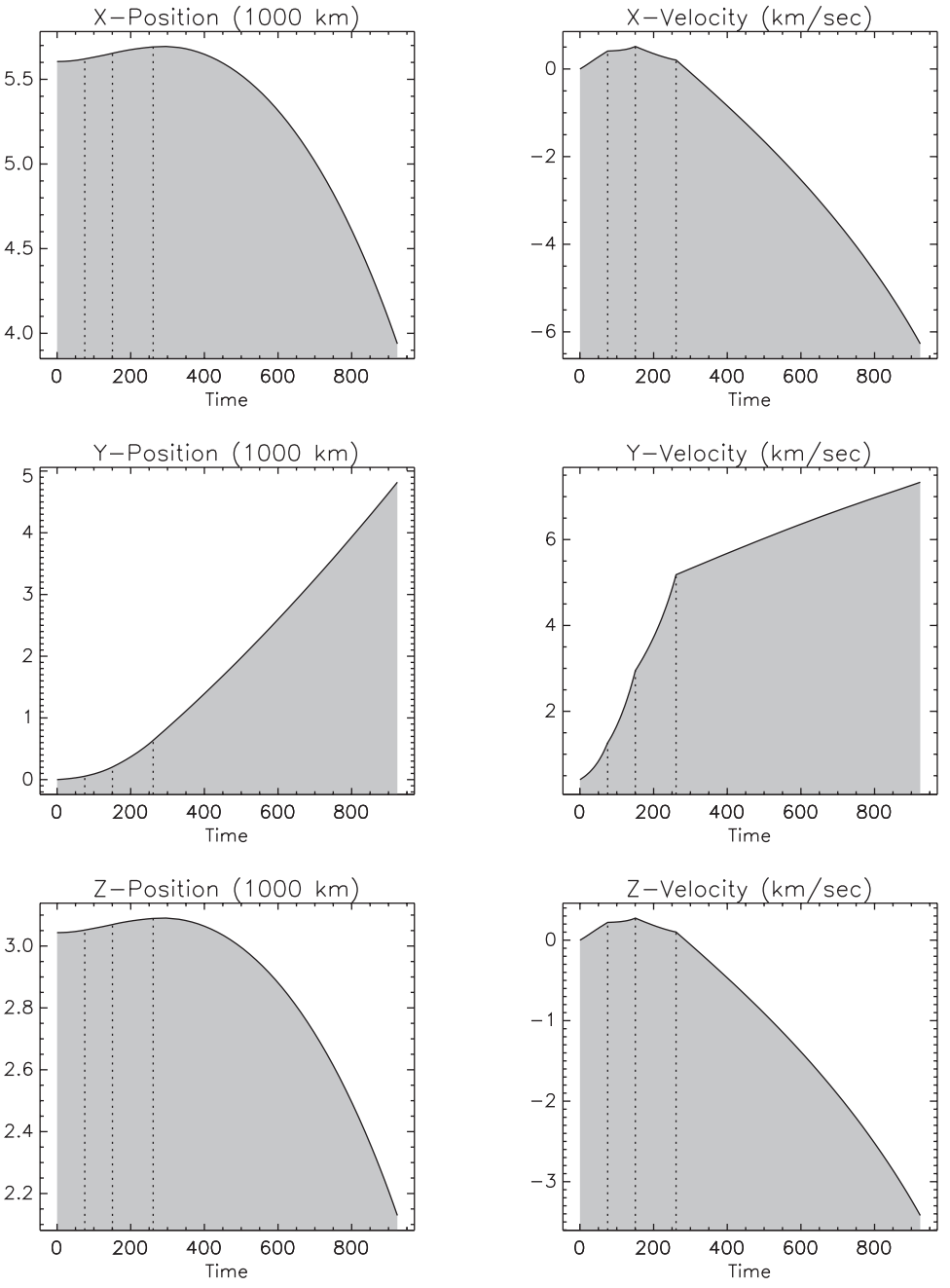


Figure 6.53. *Delta* III *ascent; states* $\mathbf{r}(t)$, $\mathbf{v}(t)$.

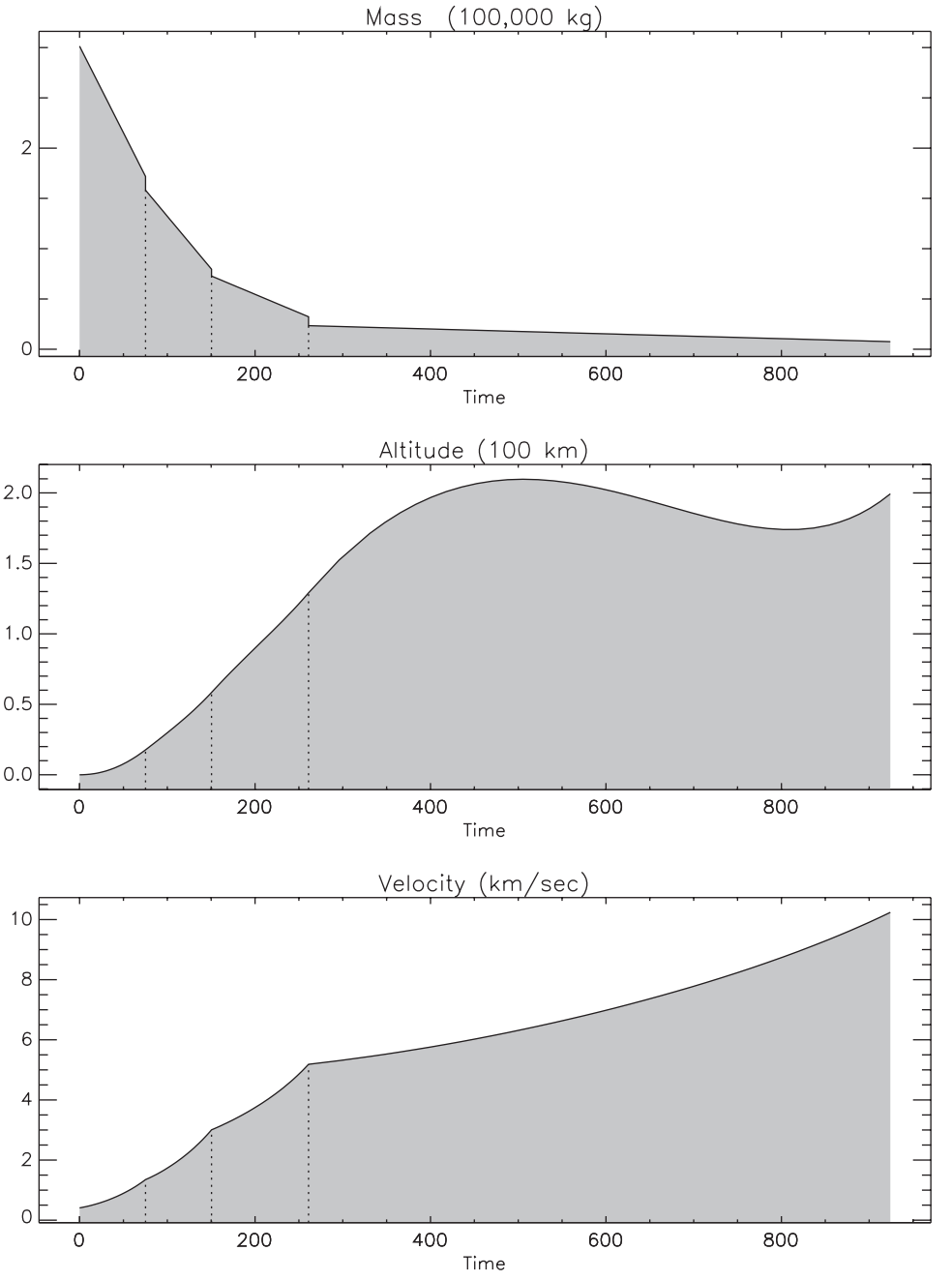


Figure 6.54. *Delta* III *ascent; mass, altitude, velocity.*

Table 6.29. *Delta* III *mesh-refinement summary.*

k	Disc.	M	n	NDOF	NQP	ϵ	Time (sec)
1	(TR,TR,TR,TR)	(10,10,10,10)	401	76	91	7.4×10^{-4}	2.00
2	(HS,TR,TR,TR)	(10,19,19,19)	761	148	6	6.9×10^{-5}	.16
3	(HC,HC,HC,HC)	(19,19,19,19)	977	292	5	2.5×10^{-6}	.37
4	(HC,HC,HC,HC)	(37,30,37,37)	1822	552	3	9.9×10^{-8}	.60
Total		141			105		3.13

6.16 A Two-Strain Tuberculosis Model

Example 6.20 TWO-STRAIN TUBERCULOSIS MODEL. In their paper Jung, Lenhart, and Feng [119] describe a model for two-strain tuberculosis treatment as follows:

> In the absence of an effective vaccine, current control programs for TB have focused on chemotherapy. The antibiotic treatment for an active TB (with drug-sensitive strain) patient requires a much longer period of time and a higher cost than that for those who are infected with sensitive TB but have not developed the disease. Lack of compliance with drug treatments not only may lead to a relapse but to the development of antibiotic resistant TB—one of the most serious public health problems facing society today. A report released by the World Health Organization warns that if countries do not act quickly to strengthen their control of TB, the multi-drug resistant strains that have cost New York City and Russia hundreds of lives and more than $1 billion each will continue to emerge in other parts of the world. The reduction in cases of drug sensitive TB can be achieved either by "case holding," which refers to activities and techniques used to ensure regularity of drug intake for a duration adequate to achieve a cure, or by "case finding," which refers to the identification (through screening, for example) of individuals latently infected with sensitive TB who are at high risk of developing the disease and who may benefit from preventive intervention. These preventive treatments will reduce the incidence (new cases per unit of time) of drug sensitive TB and hence indirectly reduce the incidence of drug resistant TB.

The dynamic model divides the host population into distinct epidemiological classes, in which the population of each class is treated as a state variable. Thus, the total population satisfies the relation

$$N = S + L_1 + I_1 + L_2 + I_2 + T, \tag{6.302}$$

where Table 6.30 describes the individual states as well as their respective initial conditions.

Table 6.30. *TB model state variables.*

Description	State	Initial Condition
Susceptible	$S(t)$	$76N/120$
Treated Effectively	$T(t)$	$1N/120$
Latent, infected with typical TB, not infectious	$L_1(t)$	$36N/120$
Latent, infected with resistant TB, not infectious	$L_2(t)$	$2N/120$
Infectious, with typical TB	$I_1(t)$	$4N/120$
Infectious, with resistant TB	$I_2(t)$	$1N/120$

The dynamic behavior of this system is described by the following system of ODEs:

$$\dot{S} = \Lambda - \beta_1 S\frac{I_1}{N} - \beta^* S\frac{I_2}{N} - \mu S, \tag{6.303}$$

$$\dot{T} = u_1 r_1 L_1 - \mu T + (1-(1-u_2)(p+q))r_2 I_1 - \beta_2 T\frac{I_1}{N} - \beta^* T\frac{I_2}{N}, \tag{6.304}$$

$$\dot{L}_1 = \beta_1 S\frac{I_1}{N} - (\mu+k_1)L_1 - u_1 r_1 L_1 + (1-u_2)pr_2 I_1 + \beta_2 T\frac{I_1}{N} - \beta^* L_1\frac{I_2}{N}, \tag{6.305}$$

$$\dot{L}_2 = (1-u_2)qr_2 I_1 - (\mu+k_2)L_2 + \beta^*(S+L_1+T)\frac{I_2}{N}, \tag{6.306}$$

$$\dot{I}_1 = k_1 L_1 - (\mu+d_1)I_1 - r_2 I_1, \tag{6.307}$$

$$\dot{I}_2 = k_2 L_2 - (\mu+d_2)I_2. \tag{6.308}$$

The dynamic model incorporates two variables $u_1(t)$ and $u_2(t)$, referred to as "case finding" and "case holding" controls, respectively. The "case finding" control represents the fraction of typical TB latent individuals that are identified and put under treatment. The effort that prevents failure of the treatment of the typical TB infectious individuals appears as the coefficient $1-u_2(t)$. Thus when the "case holding" control $u_2(t)$ is near 1, the implementation costs are high, but there is low treatment failure. Since the goal of the model is to reduce the latent and infectious groups with resistant-strain TB, while also keeping treatment costs low, they propose minimizing the composite objective

$$F = \int_0^{t_F}\left[L_2 + I_2 + \frac{1}{2}B_1 u_1^2 + \frac{1}{2}B_2 u_2^2\right]dt. \tag{6.309}$$

The control variables are bounded $.05 \le u_k(t) \le .95$ and the final time is fixed $t_F = 5$ (years). For the numerical results they also assume the total population N is constant, so $\Lambda = \mu N$, and $d_1 = d_2 = 0$. Finally, the weight factors $B_1 = 50$ and $B_2 = 500$ emphasize the cost of holding patients in treatment in comparison to screening and finding them in the first place. Table 6.31 summarizes the various parameters used in the model.

Table 6.31. *TB model parameters.*

Infection rate (susceptible)	β_1	13
Infection rate (treated)	β_2	13
Per capita natural death rate	μ	.0143
Per capita death rate induced by typical TB	d_1	0
Per capita death rate induced by resistant TB	d_2	0
Rate individual in L_1 becomes infectious	k_1	.5
Rate individual in L_2 becomes infectious	k_2	1
Treatment rate for individual with latent, typical TB	r_1	2
Treatment rate for individual with infectious, typical TB	r_2	1
Fraction of I_1 not completing treatment	p	.4
Fraction of I_2 not completing treatment	q	.1
Total population $N = S+L_1+I_1+L_2+I_2+T$	N	30000
Infection rate (uninfected)	β^*	.029
Weight factor of "case finding" control u_1	B_1	50
Weight factor of "case holding" control u_2	B_2	500
Recruitment Rate	Λ	μN

The optimal solution obtained using $\mathbb{SOCS}$ is illustrated in Figure 6.55 and yields an optimal objective function value of $F^* = 5.1520731 \times 10^3$. The total number of individuals infected with resistant TB at the final time is $L_2 + I_2 = 241.57031 + 881.34819 \approx 1123$.

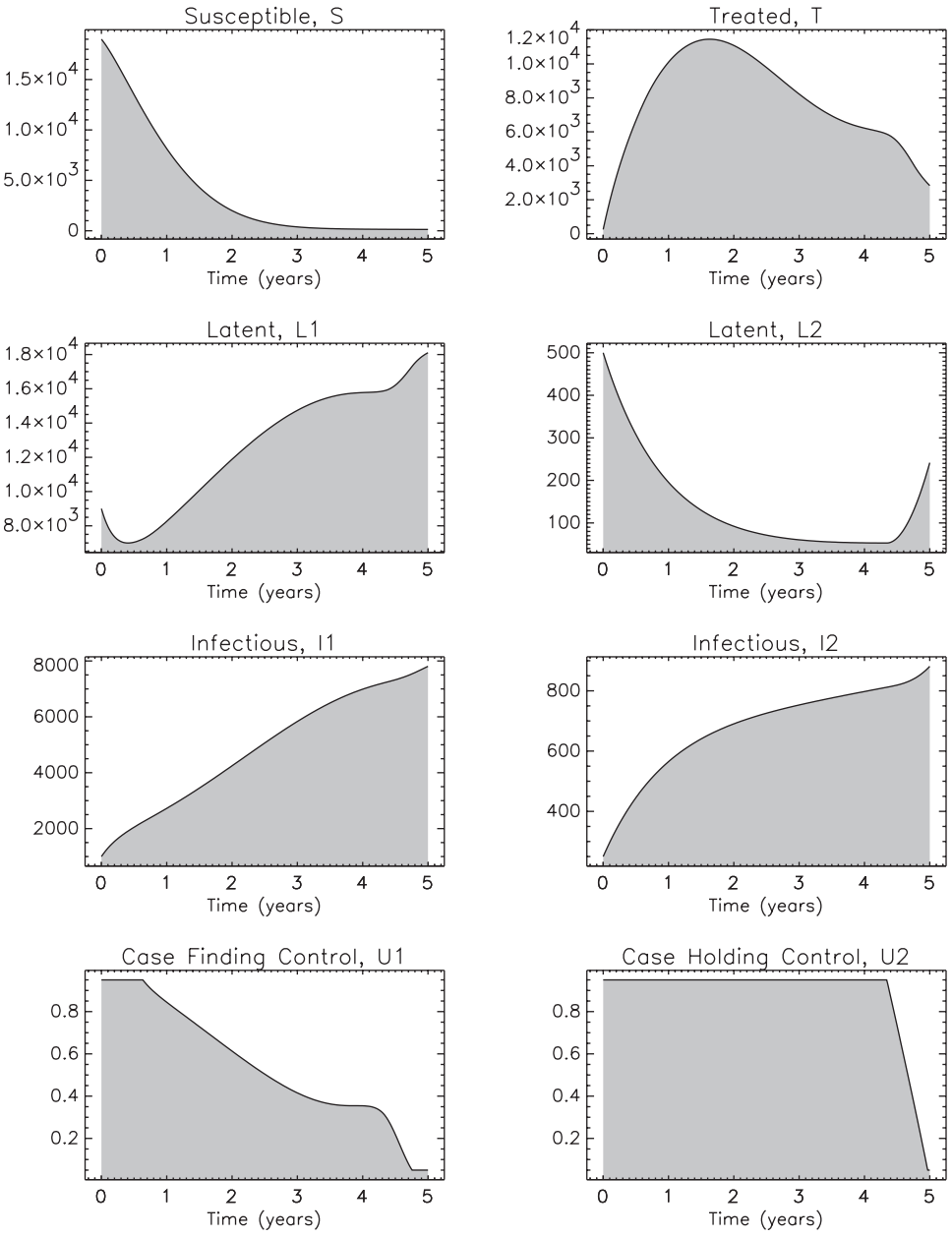


Figure 6.55. *Optimal state and control.*

6.17 Tumor Anti-angiogenesis

Example 6.21 TUMOR ANTI-ANGIOGENESIS. In their paper Ledzewicz and Schättler [128] state

> ... the search for cancer treatment methods that would circumvent the problem of drug resistance is of tantamount importance. One such approach is tumour anti-angiogenesis.
>
> A growing tumour, after it reaches just a few millimetres in size, no longer can rely on blood vessels of the host for its supply of nutrients, but it needs to develop its own vascular system for blood supply. In this process, called *angiogenesis*, there is a bi-directional signaling between tumour cells and endothelial cells: tumour cells produce vascular endothelial growth factor (VEGF) to stimulate endothelial cell growth; endothelial cells in turn provide the lining for the newly forming blood vessels that supply nutrients to the tumour and thus sustain tumour growth. But endothelial cells also have receptors that make them sensitive to inhibitors of inducers of angiogenesis like, for example, endostatin, and pharmacological therapies typically target the growth factor VEGF trying to impede the development of new blood vessels and capillaries and thus block its growth. The tumour, deprived of necessary nutrition, regresses. Since the treatment targets normal cells, no occurrence of drug resistance has been reported in laboratory studies ... For this reason tumour anti-angiogenesis has been called a therapy resistant to resistance which provides a new hope in the treatment of tumour-type cancers.

In [128] the interaction between tumor cells and endothelial cells is described by a system with the primary tumor volume, p, and the carrying capacity of the vascular, q, as variables. The control which corresponds to the angiogenic dose rate is bounded $0 \le u(t) \le a$, and a constraint on the total anti-angiogenic treatment administered

$$\int_0^{t_F} u(t)dt \le A$$

is also imposed. An additional variable y equal to the integral is introduced so that the constraint can be written as $y(t_F) \le A$. The resulting dynamics are thus given by

$$\dot{p} = -\xi p \ln\left(\frac{p}{q}\right), \tag{6.310}$$

$$\dot{q} = q\left[b - (\mu + dp^{\frac{2}{3}} + Gu)\right], \tag{6.311}$$

$$\dot{y} = u \tag{6.312}$$

for $0 \le t \le t_F$, with initial conditions $p(0) = p_0$, $q(0) = q_0$, and $y(0) = 0$. The constant ξ denotes a tumor growth parameter, while G is a constant that represents the anti-angiogenic killing parameter. The "birth" rate is modeled by the constant b, the "death" rate is given by d, and μ is a small parameter introduced to describe the loss of endothelial cells due to natural causes. The system has an asymptotically stable focus at $\bar{p} = \bar{q} = \left[(b-\mu)/d\right]^{3/2}$, and consequently the domain is restricted to $0 < p \le \bar{p}$ and $0 < q \le \bar{q}$. Ledzewicz and Schättler [128] demonstrate the problem is well-posed provided the initial conditions satisfy $p_0 \ge q_0$. For our illustration we choose $p_0 = \bar{p}/2$ and $q_0 = \bar{q}/4$ with the remaining parameters defined in Table 6.32. The final time t_F is free and the goal is to minimize the size of the tumor at the final time, i.e., minimize $p(t_F)$.

Table 6.32. *Tumor model parameters.*

Parameter	Value
ξ	0.084 per day
b	5.85 per day
d	0.00873 per mm^2 per day
G	0.15 kg per mg of dose per day
μ	0.02 per day
a	75
A	15

Because the control appears linearly in the differential equations the solution is "bang-bang," and a complete analysis of the necessary conditions is given in [128]. In particular, the adjoint equations are

$$\dot{\lambda}_1 = \xi\lambda_1\left[\ln\left(\frac{p}{q}\right)+1\right]+\frac{2}{3}\lambda_2 dqp^{-\frac{1}{3}}, \tag{6.313}$$

$$\dot{\lambda}_2 = -\xi\lambda_1\frac{p}{q}+\lambda_2\left[b-(\mu+dp^{\frac{2}{3}}+Gu)\right], \tag{6.314}$$

$$\dot{\lambda}_3 = 0 \tag{6.315}$$

with Hamiltonian function

$$H = -\lambda_1\xi p\ln\left(\frac{p}{q}\right)+\lambda_2 q\left[b-(\mu+dp^{\frac{2}{3}}+Gu)\right]+\lambda_3 u. \tag{6.316}$$

The Hamiltonian is minimized by choosing the optimal control given by

$$u^*(t) = \begin{cases} 0 & \text{if } \Phi(t) > 0, \\ a & \text{if } \Phi(t) < 0, \end{cases} \tag{6.317}$$

where the switching function is

$$\Phi(t) = \lambda_3(t) - \lambda_2(t)Gq(t). \tag{6.318}$$

For our illustration there is a single switch time t_s that occurs when the switching function is zero, and as such it is natural to consider a formulation with two distinct phases. During phase 1, the control is at its upper bound $u(t) = a$ and during phase 2 at the lower bound $u(t) = 0$. The indirect formulation is complete when the remaining boundary conditions

$$\lambda_1(t_F) = 1, \qquad H(t_F) = 0, \tag{6.319}$$

$$\lambda_2(t_F) = 0, \qquad \Phi(t_s) = 0 \tag{6.320}$$

are satisfied.

The simplest way to compute numerical results for this problem is to apply the $\mathbb{SOCS}$ direct transcription method. This approach does not require knowledge of the switching structure, does not require implementation of the indirect necessary conditions, and does not require initial estimates for the adjoint variables. Table 6.33 briefly summarizes the mesh-refinement history and solution time needed to reach the requested accuracy of $\epsilon < 10^{-7}$.

A second alternative is to incorporate knowledge of the phase structure within the direct formulation. Specifically, one can formulate a problem with two phases. During

Table 6.33. *Mesh refinement—direct formulation.*

k	M	ϵ	Time (sec)
1	60	4×10^{-5}	.11
2	119	2×10^{-6}	1.0
3	119	7×10^{-7}	1.2
4	124	5×10^{-8}	.67
Total	124		2.98

Table 6.34. *Method comparison.*

Method	$p^*(t_F)$	t_F^*
One-Phase Direct	7.5716831×10^3	1.1954773
Two-Phase Direct	7.5716700×10^3	1.1963497
Two-Phase Indirect	7.5716701×10^3	1.1963496

phase one the control is $u(t) = a$, and during phase two $u(t) = 0$. The switch time t_s is treated as free. This approach incorporates knowledge about the discontinuous behavior in $u(t)$ without explicitly imposing the switching conditions. Using the solution from the single phase formulation provides an excellent initial guess for the two phase problem, and convergence is extremely rapid. Finally the third alternative is to derive the adjoint and transversality conditions and then solve the indirect formulation using either an indirect collocation or indirect shooting method. The discrete adjoint estimates from the second formulation provide an excellent guess for this approach, and again converged results are obtained easily. Table 6.34 presents a comparison of the results obtained by each method, all with the requested accuracy of $\epsilon < 10^{-7}$. Clearly the two-phase results are consistent to seven significant figures, whereas the one-phase direct result differs slightly in the seventh digit. This small inaccuracy can be attributed to the method used to represent the optimal bang-bang control. In particular when two phases are used, the step discontinuity in the control can be modeled accurately. In contrast, when the problem is treated using a single phase, the control history is continuous by construction, and the mesh must be refined in order to approximate the step discontinuity. Figure 6.56 illustrates the continuous control approximation (solid line) using the one-phase direct method, compared with the discon-

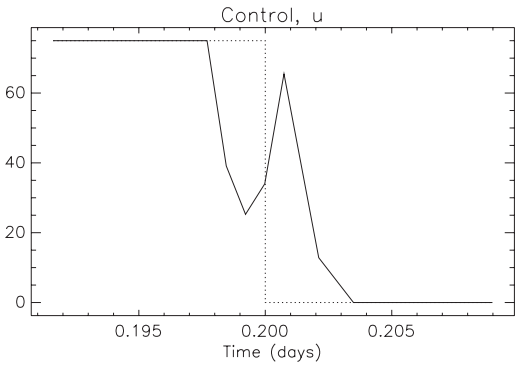


Figure 6.56. *Control.*

tinuous (dotted line) control from the two-phase method. Figure 6.57 illustrates the optimal states, adjoints, and switching function.

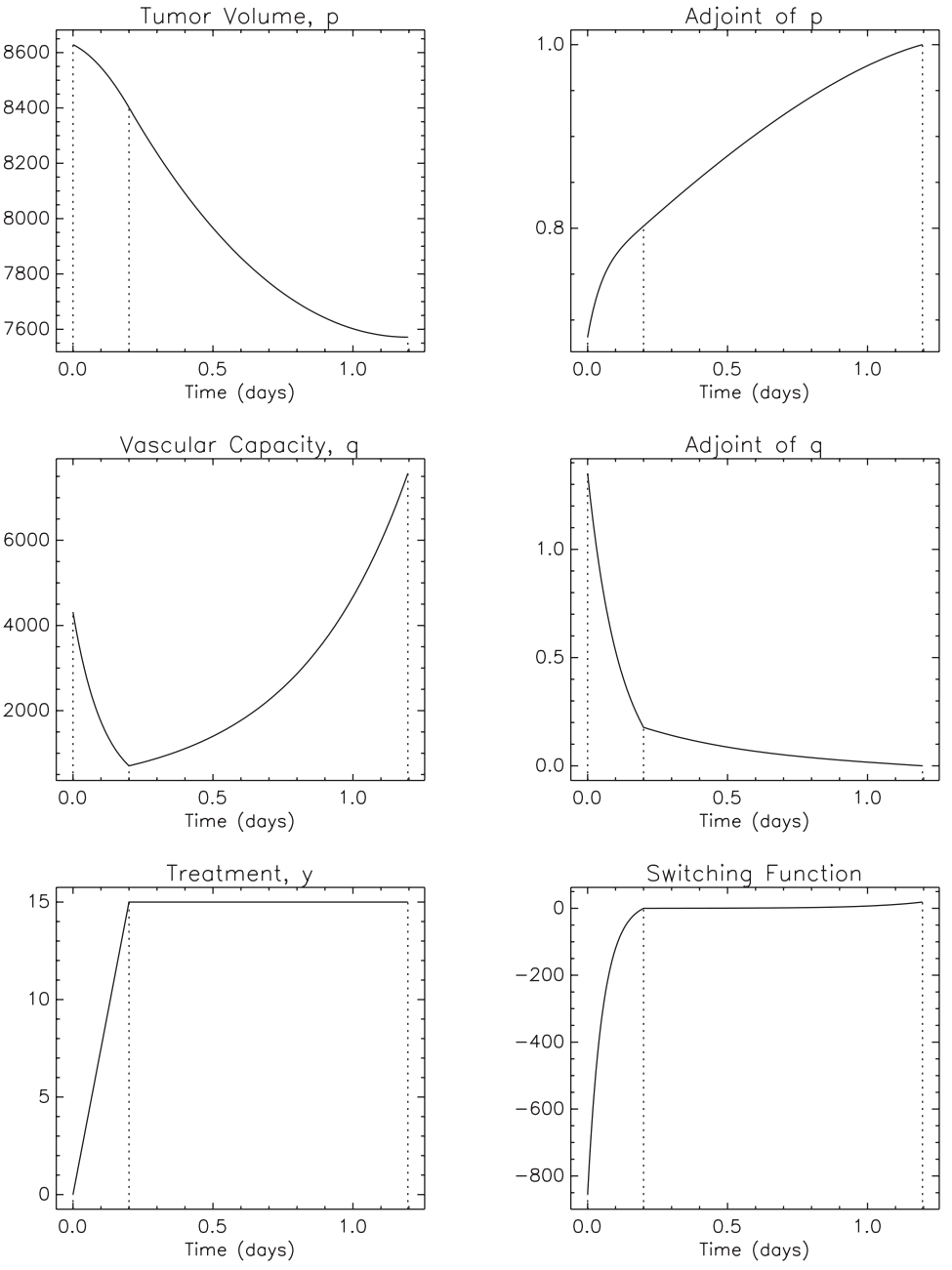


Figure 6.57. *States.*

This problem illustrates two important points. First, if the control (or state) is discontinuous at the solution, it is important to formulate the problem using multiple phases. The multiphase formulation permits an accurate representation of discontinuous behavior. Second, when mesh refinement is used with a direct method there is no loss of accuracy compared to an indirect method. In short, a direct and an indirect method have comparable accuracy. Thus, as a practical matter, there is no reason to derive the adjoint equations, unless they are needed for some other purpose.