JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 48, No. 3, May–June 2011

Direct Trajectory Optimization Using a Variable Low-Order Adaptive Pseudospectral Method

Christopher L. Darby,* William W. Hager,† and Anil V. Rao‡
University of Florida, Gainesville, Florida 32611

DOI: 10.2514/1.52136

A variable-order adaptive pseudospectral method is presented for solving optimal control problems. The method developed in this paper adjusts both the mesh spacing and the degree of the polynomial on each mesh interval until a specified error tolerance is satisfied. In regions of relatively high curvature, convergence is achieved by refining the mesh, while in regions of relatively low curvature, convergence is achieved by increasing the degree of the polynomial. An efficient iterative method is then described for accurately solving a general nonlinear optimal control problem. Using four examples, the adaptive pseudospectral method described in this paper is shown to be more efficient than either a global pseudospectral method or a fixed-order method.

Nomenclature

$\mathbf{C}$	=	path constraint function
D	=	$N \times (N+1)$ Radau pseudospectral differentiation matrix
E	=	maximum absolute solution error
F_d	=	magnitude of drag force, N
F_g	=	magnitude of gravity force, N
F_l	=	magnitude of lift force, N
F_t	=	magnitude of thrust force, N
g	=	integrand of cost functional
h	=	altitude
I	=	number of grid refinement iterations
J	=	continuous-time cost functional
K	=	number of mesh intervals
L	=	magnitude of lift force, N
$\mathcal{L}(\tau)$	=	Lagrange polynomial on time domain $\tau \in [-1, +1]$
m	=	dimension of continuous-time control
N	=	polynomial degree
N_a	=	number of approximation points
N_c	=	number of collocation points
N_k	=	polynomial degree in mesh interval k
N_z	=	number of nonzero constraint Jacobian entries
n	=	dimension of continuous-time state
r	=	geocentric radius, m
R_e	=	equatorial radius of Earth, m
s	=	dimension of path constraint function
T	=	CPU time, s
t	=	time on time interval $t \in [t_0, t_f]$
t_f	=	terminal time
t_0	=	initial time
t_s^*	=	control switch time
$\mathbf{X}(\tau)$	=	state approximation on time domain $\tau \in [-1, +1]$
$x(t)$	=	component of position
$\mathbf{x}(t)$	=	state on time domain $t \in [t_0, t_f]$
$\mathbf{x}(\tau)$	=	state on time domain $\tau \in [-1, +1]$
$y(t)$	=	component of position
$\mathbf{y}(t)$	=	general vector function on time domain $t \in [t_0, t_f]$
$\mathbf{y}(\tau)$	=	general vector function on time domain $\tau \in [-1, +1]$
$u_{\max}$	=	maximum allowable value of control
$u_{\min}$	=	minimum allowable value of control
$\mathbf{U}(\tau)$	=	control approximation on time domain $\tau \in [-1, +1]$
$u(t)$	=	control on time domain $t \in [t_0, t_f]$
$\mathbf{u}(t)$	=	control on time domain $t \in [t_0, t_f]$
$\mathbf{u}(\tau)$	=	control on time domain $\tau \in [-1, +1]$
v	=	speed
w_j	=	jth Legendre–Gauss–Radau quadrature weight
α	=	angle of attack, rad and deg
γ	=	flight-path angle, rad and deg
ϵ	=	accuracy tolerance
η	=	load factor
θ	=	orientation angle or longitude, rad
κ	=	trajectory curvature
μ	=	Earth gravitational parameter, m^3/s^2
σ	=	bank angle, rad
τ	=	time on time interval $\tau \in [-1, +1]$
Φ	=	Mayer cost
ϕ	=	latitude, rad and deg
$\boldsymbol{\phi}$	=	boundary condition function
ψ	=	azimuth angle, rad and deg

I. Introduction

OVER the past two decades, direct collocation methods have become popular in the numerical solution of nonlinear optimal control problems. In a direct collocation method, the state is approximated using a set of trial (basis) functions and the dynamics are collocated at specified set of points in the time interval. Most commonly, direct collocation methods for optimal control are employed using so-called h methods where a fixed low-degree polynomial (e.g., second degree or third degree) state approximation is used and the problem is divided into many intervals. Convergence of the numerical discretization is then achieved by increasing the number of mesh intervals [1–3]. Grid refinement techniques are commonly used to obtain a specified solution accuracy by increasing the number of mesh intervals in regions of the trajectory where the errors are largest. Excellent examples of h methods for solving optimal control problems are given in [2–6].

In recent years, pseudospectral methods for solving optimal control problems have increased in popularity [7–19]. In a pseudospectral method, the collocation points are based on accurate quadrature rules and the basis functions are typically Chebyshev or Lagrange polynomials. In contrast to an h method, a pseudospectral

Presented in an abbreviated format as Paper 2010-8272 at the 2010 AIAA/AAS Astrodynamics Specialist Conference, Toronto, Ontario, Canada, 2–5 August 2010; received 23 August 2010; revision received 18 November 2010; accepted for publication 19 November 2010.

*Ph.D. Student. Department of Mechanical and Aerospace Engineering; cdarby@ufl.edu.

†Professor. Department of Mathematics; hager@ufl.edu.

‡Assistant Professor. Department of Mechanical and Aerospace Engineering; anilvrao@ufl.edu. Associate Fellow AIAA (Corresponding Author).

method is typically employed as a p method where a single mesh interval is used, and convergence is achieved by increasing the degree, p, of the polynomial. For problems where the solutions are infinitely smooth and well behaved, a pseudospectral method has a simple structure and converges at an exponential rate [20–22]. The most well-developed pseudospectral methods are the Gauss pseudospectral method [12,13], the Radau pseudospectral method (RPM) [17–19], and the Lobatto pseudospectral method [7].

While pseudospectral methods have typically been applied as p methods [7,8,12,13,18,19], relying on convergence using global polynomials has several limitations. First, even for smooth problems, an accurate approximation may require an unreasonably large-degree global polynomial. In addition, the convergence rate of a p method on a problem for which the formulation or solution is nonsmooth may be extremely slow, resulting in a poor approximation even when a high-degree polynomial is used. A second limitation of a p method is that the use of a high-degree global polynomial results in a nonlinear programming problem (NLP) for which the density grows quickly as a function of the number of NLP variables. This growth in the NLP density is due to the dense blocks that arise from the global pseudospectral differentiation matrix. Thus, a p method becomes computationally intractable for problems when an excessively large-degree polynomial is required. While h methods are computationally more tractable than p methods, they may require a large number of mesh intervals in order to achieve an acceptable accuracy because exponential convergence is lost using an h method. Moreover, when the NLP is sufficiently large, it may be difficult to compute a solution.

To simultaneously improve accuracy and computational efficiency using pseudospectral methods, we combine the best features of both an h method and a p method to form a so-called hp method. As its name implies, an hp method is one where the number of mesh intervals, the mesh interval widths, and the polynomial degree in each mesh interval is determined algorithmically. Previously, hp methods have been developed in the context of finite elements in mechanics and spectral methods in fluid dynamics. In particular, [23–27] describe the mathematical properties of h, p, and hp methods for finite elements. Galvao et al. [28] showed the application of an hp-adaptive least-squares spectral element method (LS-SEM) for solving hyperbolic partial differential equations. Heinrichs [29] developed an adaptive spectral least-squares collocation scheme for Burgers's equation. Dorao and Jakobsen [30] showed how to apply of an hp-adaptive LS-SEM to the population balance equation, while Dorao et al. [31] developed an hp-adaptive spectral element solver for reactor modeling. An overview of hp-adaptive spectral element methods for solving problems in computational fluid dynamics can be found in [32].

In this paper, we develop a new hp-adaptive pseudospectral method for solving optimal control problems. In the method developed in this paper, the accuracy of the solution is improved either by increasing the degree of the polynomial within a mesh interval or by refining the mesh. The decision to increase the polynomial degree or refine the mesh is based on the relative curvature in each mesh interval. If the ratio between the maximum curvature and the average curvature is sufficiently large on an interval, then the accuracy is increased by refining the mesh and by using low-degree polynomials in the newly created mesh intervals. Otherwise, the accuracy is improved by increasing the degree of approximating polynomial within a mesh interval. The method is demonstrated on four examples of varying complexity and is found to be a viable method for efficiently and accurately solving complex optimal control problems.

It is noted that [33] also develops an hp-adaptive pseudospectral method for solving optimal control problems. The method of [33] starts with a global approximation. The accuracy is then improved by increasing the degree of the polynomial within a mesh interval until it is deemed necessary to refine the mesh. The mesh is refined only when exponential convergence is lost within an existing mesh interval. As a result, the method of [33] more closely resembles a p method because it can result in relatively large-degree polynomials within a mesh interval. On the other hand, the method of this paper starts with a coarse mesh and uses low-degree polynomials in each mesh interval. The degree of the polynomial within a mesh interval is then increased only if the solution in that mesh interval is sufficiently smooth. Consequently, in the algorithm developed in this paper more closely resembles an h method because the degree of the approximating polynomial in a mesh interval is kept relatively small. The method of [33] may be preferable over the method of this paper for problems for which the solutions are known a priori to be smooth with the exception of a few isolated points, while the method of this paper is preferable for problems for which the solution structure is not known a priori.

We also compare our method to [3], where low-degree Runge–Kutta intervals are used. In [3], the mesh is refined globally based on the integral of the curvature. In our hp scheme, on the other hand, the refinement technique of [3] is applied locally to each mesh interval where refinement is deemed necessary. As a result, mesh interval changes are more local in the method of this paper as compared with the method of [3]. Furthermore, the algorithm of this paper allows for a different degree polynomial approximation in each mesh interval. Finally, the growth rate of the mesh in our algorithm is a function of the error in the current solution and the desired user prescribed accuracy.

This paper is organized as follows. In Sec. II, we define the Bolza optimal control problem of interest in this paper. In Sec. III, we formulate a multiple-interval discretization of the Bolza optimal control problem using the RPM. In Sec. IV, we provide two motivating examples for the use of an hp-adaptive pseudospectral method. In Sec. V, we present our variable-order hp-adaptive pseudospectral method. In Sec. VI, we provide four examples of the method using optimal control problems of varying complexity. Finally, in Secs. VII and VIII, we provide a discussion of the results and conclusions.

II. Bolza Optimal Control Problem

Consider the following fairly general optimal control problem in Bolza form. Minimize the cost functional

$$J = \Phi[\mathbf{x}(t_0), t_0, \mathbf{x}(t_0), t_f] + \int_{t_0}^{t_f} g[\mathbf{x}(t), \mathbf{u}(t), t]\,\mathrm{d}t \tag{1}$$

subject to the dynamic constraints

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{2}$$

the inequality path constraints

$$\mathbf{C}[\mathbf{x}(t), \mathbf{u}(t), t] \leq \mathbf{0} \tag{3}$$

and the boundary conditions (i.e., the event constraints)

$$\phi[\mathbf{x}(t_0), t_0, \mathbf{x}(t_0), t_f] = \mathbf{0} \tag{4}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^m$ is the control, $\mathbf{C}[\mathbf{x}(t), \mathbf{u}(t), t] \in \mathbb{R}^s$ represents the control and state constraints, and t is time. For convenience, all quantities will be treated as row vectors; that is, if $\mathbf{y}(t) \in \mathbb{R}^q$, then $\mathbf{y}(t) \equiv [\,y_1(t), \quad \cdots, \quad y_q(t)\,]$.

Suppose now that the time interval $t \in [t_0, t_f]$ is divided into a mesh consisting of K mesh intervals $[t_{k-1}, t_k]$, $k = 1, \ldots, K$, where $(t_0, \ldots, t_K)$ are the mesh points; the mesh points have the property that $t_0 < t_1 < t_2 < \cdots < t_K = t_f$. In each mesh interval $t \in [t_{k-1}, t_k]$, we make the change of variables

$$\tau = \frac{2t - (t_k + t_{k-1})}{t_k - t_{k-1}}, \qquad (t_{k-1} < t_k) \tag{5}$$

Using the change of variables given in Eq. (5), the interval $t \in [t_{k-1}, t_k]$ is transformed to $\tau \in [-1, +1]$. Moreover, we have

$$\frac{\mathrm{d}\tau}{\mathrm{d}t} = \frac{2}{t_k - t_{k-1}}, \qquad (k = 1, \ldots, K) \tag{6}$$

Next, let $\mathbf{x}^{(k)}(\tau)$ and $\mathbf{u}^{(k)}(\tau)$ be the state and control in the kth mesh interval as a function of $\tau \in [-1, +1]$. Using Eq. (6), the Bolza optimal control problem of Eqs. (1–4) can be written as follows. First, the cost functional of Eq. (1) can be written as

$$J = \Phi[\mathbf{x}^{(1)}(-1), t_0, \mathbf{x}^{(K)}(+1), t_f] + \sum_{k=1}^{K} \frac{t_k - t_{k-1}}{2} \int_{-1}^{+1} g[\mathbf{x}^{(k)}(\tau), \mathbf{u}^{(k)}(\tau), \tau; t_{k-1}, t_k]\, d\tau \quad (7)$$

Next, the dynamics of Eq. (2) and the path constraints of Eq. (3) are given in terms of τ in mesh interval $k \in [1, \ldots, K]$, respectively, as

$$\frac{d\mathbf{x}^{(k)}(\tau)}{d\tau} = \frac{t_k - t_{k-1}}{2} \mathbf{f}[\mathbf{x}^{(k)}(\tau), \mathbf{u}^{(k)}(\tau), \tau; t_{k-1}, t_k] \quad (8)$$

$$\mathbf{0} \geq \mathbf{C}[\mathbf{x}^{(k)}(\tau), \mathbf{u}^{(k)}(\tau), \tau; t_{k-1}, t_k] \quad (9)$$

Furthermore, the boundary conditions of Eq. (4) are given as

$$\boldsymbol{\phi}\,[\mathbf{x}^{(1)}(-1), t_0, \mathbf{x}^{(K)}(+1), t_f] = \mathbf{0} \quad (10)$$

Finally, it is required that the state be continuous at the interface of mesh intervals; that is, $\mathbf{x}(t_k^-) = \mathbf{x}(t_k^+)$, $k = 1, \ldots, K-1$.

III. Radau Pseudospectral Method

The hp method in this paper is developed by discretizing the multiple-interval form of the Bolza optimal control problem, given in Sec. II, using the previously developed RPM, as described in [18] While we choose the RPM to discretize the optimal control problem, with only slight modifications, the hp-adaptive method described this paper can be developed using other pseudospectral methods (e.g., the Gauss [12,13,15] or the Lobatto [7] pseudospectral method). An advantage of using the Radau scheme is that the continuity conditions $\mathbf{x}(t_k^-) = \mathbf{x}(t_k^+)$ across mesh points are particularly easy to implement.

In the RPM, the state of the continuous Bolza problem is approximated within each mesh interval $k \in [1, \ldots, K]$ as

$$\mathbf{x}^{(k)}(\tau) \approx \mathbf{X}^{(k)}(\tau) = \sum_{j=1}^{N_k+1} \mathbf{X}_j^{(k)} \mathcal{L}_j^{(k)}(\tau), \qquad \mathcal{L}_j^{(k)}(\tau) = \prod_{\substack{l=1 \\ l \neq j}}^{N_k+1} \frac{\tau - \tau_l^{(k)}}{\tau_j - \tau_l^{(k)}} \quad (11)$$

where $\tau \in [-1, +1]$, $\mathcal{L}_j^{(k)}(\tau)$, $j = 1, \ldots, N_k + 1$, is a basis of Lagrange polynomials, $(\tau_1^{(k)}, \ldots, \tau_{N_k}^{(k)})$ are the Legendre–Gauss–Radau (LGR) [18,34] collocation points in mesh interval k, and $\tau_{N_k+1}^{(k)} = +1$ is a noncollocated point. The cost functional of Eq. (7) is then approximated using a multiple-interval LGR quadrature as

$$J \approx \Phi(\mathbf{X}_1^{(1)}, t_0, \mathbf{X}_{N_K+1}^{(K)}, t_K) + \sum_{k=1}^{K} \sum_{j=1}^{N_k} \left(\frac{t_k - t_{k-1}}{2} \right) w_j^{(k)} g(\mathbf{X}_j^{(k)}, \mathbf{U}_j^{(k)}, \tau_j^{(k)}; t_{k-1}, t_k) \quad (12)$$

where $\mathbf{U}_i^{(k)}$, $i = 1, \ldots, N_k$, are the approximations of the control at the N_k LGR points in the kth mesh interval, $\mathbf{X}_1^{(1)}$ is the approximation of $\mathbf{x}(t_0)$, and $\mathbf{X}_{N_K+1}^{(K)}$ is the approximation of $\mathbf{x}(t_f)$. Differentiating $\mathbf{X}^{(k)}(\tau)$ in Eq. (11) with respect to τ, we obtain

$$\frac{d\mathbf{X}^{(k)}(\tau)}{d\tau} \equiv \dot{\mathbf{X}}^{(k)}(\tau) = \sum_{j=1}^{N_k+1} \mathbf{X}_j^{(k)} \dot{\mathcal{L}}_j^{(k)}(\tau) \quad (13)$$

Collocating the dynamics at the N_k LGR points using Eq. (13), we have

$$\sum_{j=1}^{N_k+1} \mathbf{X}_j^{(k)} D_{ij}^{(k)} - \frac{t_k - t_{k-1}}{2} \mathbf{f}(\mathbf{X}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_{k-1}, t_k) = \mathbf{0}, \quad (i = 1, \ldots, N_k) \quad (14)$$

where

$$D_{ij}^{(k)} = \dot{L}_j^{(k)}(\tau_i^{(k)}), \qquad (i = 1, \ldots, N_k) \qquad (j = 1, \ldots, N_k + 1) \quad (15)$$

is the $N_k \times (N_k + 1)$ Radau pseudospectral differentiation matrix [18] in the kth mesh interval. Furthermore, the path constraints of Eq. (3) in mesh interval $k \in [1, \ldots, K]$ are enforced at the N_k LGR points as

$$\mathbf{C}\,(\mathbf{X}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_{k-1}, t_k) \leq \mathbf{0}, \qquad (i = 1, \ldots, N_k) \quad (16)$$

Finally, the event constraints are approximated as

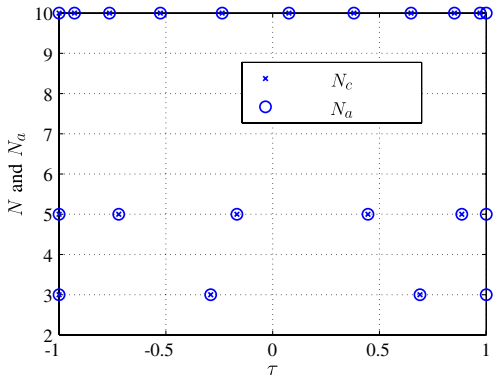
$$\boldsymbol{\phi}\,(\mathbf{X}_1^{(1)}, t_0, \mathbf{X}_{N_K+1}^{(K)}, t_K) = \mathbf{0} \quad (17)$$

Continuity across the mesh points is maintained by the condition

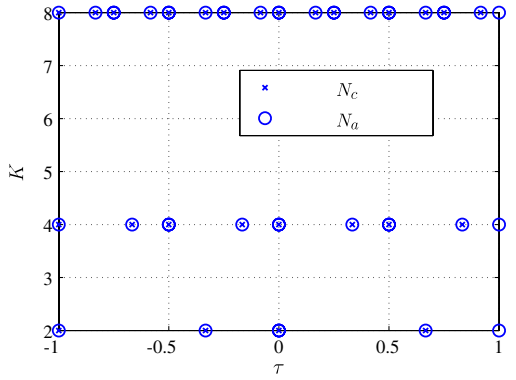
$$\mathbf{X}_{N_k+1}^{(k)} = \mathbf{X}_1^{(k+1)} \quad (18)$$

The NLP that arises from the Radau pseudospectral approximation is then to minimize the cost function of Eq. (12) subject to the algebraic constraints of Eqs. (14–18). In our computer implementation of the NLP, we use the same variable for both $\mathbf{X}_{N_k+1}^{(k)}$ and $\mathbf{X}_1^{(k+1)}$. Hence, the constraint (18) can be eliminated since it is explicitly taken into account.

It is useful to observe a few key properties of the RPM. First, when used as a single-interval p method (see Fig. 1a), the state is approximated at the N_k collocation points plus the terminal point,



a) N_c and N_a for for a p–type radau collocation method



b) N_c and N_a for an h–type radau collocation method

Fig. 1 Number of collocation points N_c and number of approximation points N_a for a p- and h-Radau collocation method.

while the control is approximated at only the N_k collocation points. For a multi-interval discretization where a Radau scheme is used for each interval in the mesh, the control is approximated at every point where the state is approximated with the single exception of the final point $t = t_f$ (see Fig. 1b). Finally, we comment on the fact that the RPM described in this section can be written equivalently in either differential or implicit integral form [18,19]. We choose to use the differential form (that is, the form described in this paper) because it has computational advantages over the integral form. See [18,19] for further details on the equivalence between the differential and integrals forms of the RPM.

IV. Motivating Examples

In this section, we consider examples that motivate the development of an hp-adaptive pseudospectral method for solving optimal control problems. The first example has a smooth solution, while the second example has a nonsmooth solution. The goal of the motivating examples is to provide insight into when an h method or a p method may be more appropriate.

A. Example 1: Problem with a Smooth Solution

Consider the following optimal control problem [35]. Minimize the cost functional

$$J = t_f \tag{19}$$

subject to the dynamic constraints

$$\dot{x} = \cos(\theta), \qquad \dot{y} = \sin(\theta), \qquad \dot{\theta} = u \tag{20}$$

and the boundary conditions

$$\begin{aligned} x(0) = 0, \qquad y(0) = 0, \qquad \theta(0) = -\pi \qquad x(t_f) = 0, \\ y(t_f) = 0, \qquad \theta(t_f) = \pi \end{aligned} \tag{21}$$

The solution to the optimal control problem of Eqs. (19–21) is

$$[x^*(t), y^*(t), \theta^*(t), u^*(t)] = [-\sin(t), -1 + \cos(t), t - \pi, 1] \tag{22}$$

where $t_f^* = 2\pi$. It is seen that the solution to the optimal control problem given in Eqs. (19–21) is smooth. We will approximate this solution using a Radau pseudospectral p method, and we will compare the error to that of a uniformly spaced second-degree Radau h method. Figures 2a and 2b show the maximum state error as a function of the number of collocation points and the number of nonzero constraint Jacobian entries, respectively, for both the p method and the second-degree h method. Because the solution to this problem is smooth, the p method converges at an exponential rate, while the second-degree h method converges at a significantly slower rate as compared with the p method. Consequently, in this example, convergence is achieved much more rapidly using a p method. In particular, for any given desired solution accuracy, the overall size of the NLP resulting from the p method is much smaller than the size from the h method, thus making the NLP of the p method much easier to solve.

B. Example 2: Problem with a Nonsmooth Solution

Consider the following optimal control problem of a soft lunar landing, as given in [36]. Minimize the cost functional

$$J = \int_0^{t_f} u \, \mathrm{d}t \tag{23}$$

subject to the dynamic constraints

$$\dot{h} = v, \qquad \dot{v} = -g + u \tag{24}$$

the boundary conditions

$$[h(0), v(0), h(t_f), v(t_f)] = (h_0, v_0, h_f, v_f) = (10, -2, 0, 0) \tag{25}$$

and the control inequality constraint

$$u_{\min} \le u \le u_{\max} \tag{26}$$

where $u_{\min} = 0$, $u_{\max} = 3$, $g = 1.5$, and t_f is free. The optimal solution to this example is

$$\begin{aligned} &(h^*(t), v^*(t), u^*(t)) \\ &= \begin{cases} (-\frac{3}{4}t^2 + v_0 t + h_0, -\frac{3}{2}t + v_0, 0), & t < t_s^* \\ \left[\frac{3}{4}t^2 + (-3t_s^* + v_0)t + \frac{3}{2}t_s^{*2} + h_0, \frac{3}{2}t + (-3t_s^* + v_0), 3\right], & t > t_s^* \end{cases} \end{aligned} \tag{27}$$

where t_s^* is

$$t_s^* = \frac{t_f^*}{2} + \frac{v_0}{3} \tag{28}$$

with

$$t_f^* = \tfrac{2}{3}v_0 + \tfrac{4}{3}\sqrt{\tfrac{1}{2}v_0^2 + \tfrac{3}{2}h_0} \tag{29}$$

For the boundary conditions given in Eq. (25), we have $t_s^* = 1.4154$ and $t_f^* = 4.1641$. As seen in Eq. (27), the optimal control $u^*(t)$ for this example is bang–bang. As a result, the control is discontinuous and the state components $h(t)$ and $v(t)$ are nonsmooth [in this case, $h(t)$ is piecewise quadratic and $v(t)$ is piecewise linear, as seen in Eq. (27)].

The convergence rates for this problem using a Radau p method and an equally spaced Radau h-2 method are now analyzed. Examining Fig. 2c, it is seen that the p method and the h-2 method converge erratically and much more slowly than was the case for example 1. Although the convergence of both methods appears to be erratic, it turns out that the asymptotic convergence rate for both methods is log linear. In other words, the error is bounded by the product of a constant with the inverse of the degree of the approximating polynomial (in the case of the p method) or the product of a constant and the inverse of the number of mesh intervals (in the case of the h method). Moreover, the convergence rate for the h method in Fig. 2c is strongly influenced by the distance between the switch time of the continuous-time optimal solution and the closest mesh point, and this distance depends in a complicated way on the number of mesh intervals.

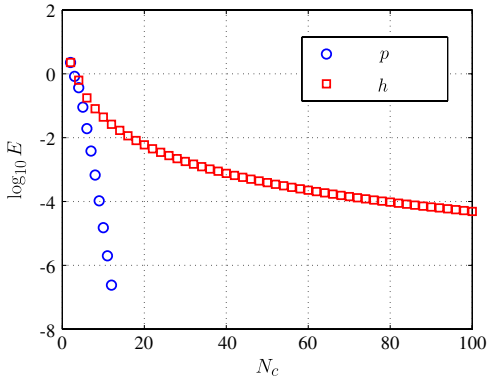
To see more clearly the underlying log linear asymptotic convergence rate of the p and h-2 methods, consider the following equivalent fixed-time formulation of the optimal control problem given in Eqs. (23–26). Minimize the cost functional

$$J = h^2(t_f) + v^2(t_f) + \int_0^{t_f} u \, \mathrm{d}t \tag{30}$$

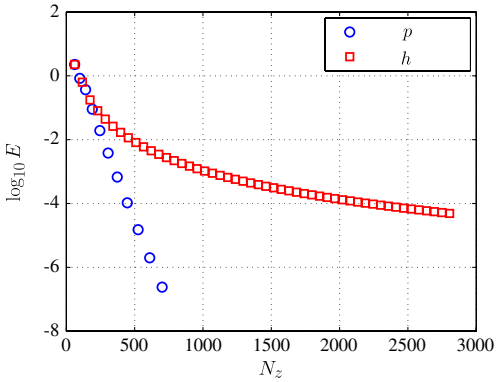
subject to the dynamic constraints given in Eq. (24), the boundary conditions

$$[h(0), v(0)] = (h_0, v_0) = (10, -2) \tag{31}$$

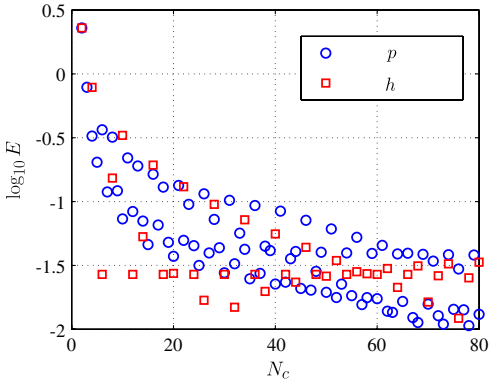
the control inequality constraint given in Eq. (26), and $t_f = t_f^*$ [where t_f^* is given by Eq. (29)]. The modified optimal control problem was solved for 2^j ($j = 1, \ldots, 8$) collocation points where, in this case, each mesh has twice the number of intervals as the previous mesh. As a result, the distance between the closest mesh point and the optimal switch point t_s^* decreases to zero monotonically as a function of j. In contrast, when all possible equally spaced meshes are considered (as shown in Fig. 2c), the distance between the closest grid point and the switching point does not decrease monotonically as a function of the number of mesh intervals. Figure 2d shows the base-10 logarithm of the maximum absolute error as a function of the base-2 logarithm of the number of collocation points. As seen in Fig. 2d, the asymptotic convergence rate of both methods is log linear. Because the asymptotic error in both the p and h-2 method are similar, their efficiency depends on the relative time it takes to solve discrete problems of similar dimensions. As seen in Figs. 2e and 2f, the



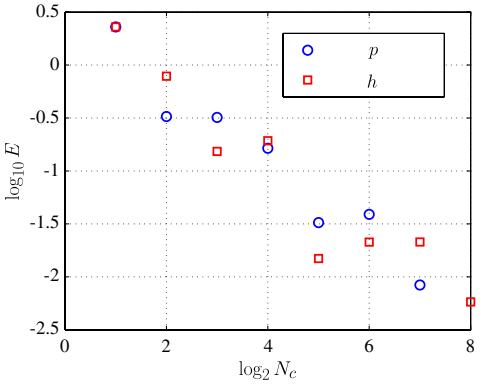
a) $\log_{10} E$ vs. N_c using a *p*–Method and a second–degree *h*–Method motivating example 1



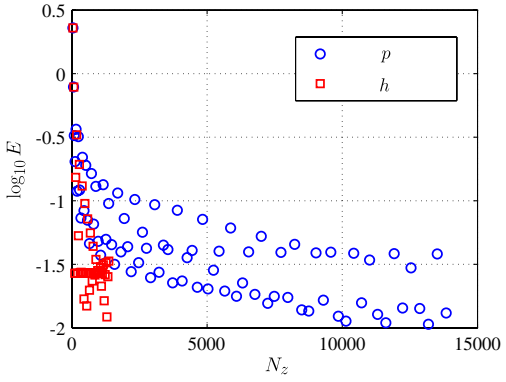
b) $\log_{10} E$ vs. N_z using a *p*–Method and a second–degree *h*–Method motivating example 1



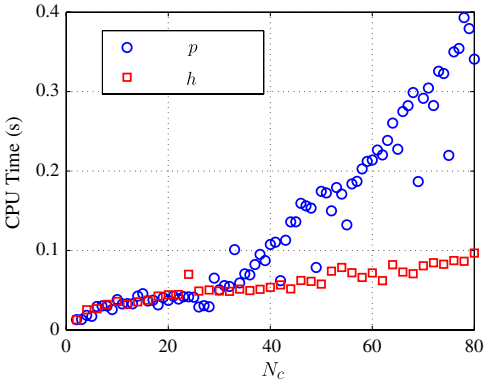
c) $\log_{10} E$ vs. N_c using a *p*–Method and a second–degree *h*–Method for the optimal control problem of eqs. (23)–(26)



d) $\log_{10} E$ vs. $\log_2 N_c$ using a *p*–Method and a second–degree *h*–Method for the alternate form of motivating example 2



e) $\log_{10} E$ vs. N_z using a *p*–Method and a second–degree *h*–Method for the motivating example 2



e) T vs. N_c using a *p*–Method and a second–degree *h*–Method for motivating example 2

Fig. 2 *h*-method and *p*-method errors for motivating examples 1 and 2.

sparsity and the solution time of the h-2 method grow much more slowly than the sparsity and solution time of the p method. Consequently, for this example, where the optimal solution has a bang–bang control, the h-2 method is much more computationally efficient.

C. Discussion of Motivating Examples

The results of the motivating examples point to the fact that, in general, the character of the solution to an optimal control problem will change in different regions of the solution. In example 1, it was seen that rapid convergence was achieved simply by increasing the degree of the polynomial. On the other hand, in example 2, it was seen that when the degree of the approximating polynomial is fixed, the rate of convergence can be improved by placing the mesh points at the appropriate locations. The solution of a general optimal control problem will require a combination of choosing the appropriate polynomial degree and the appropriate locations of the mesh intervals. In a region where the solution is well behaved, it is expected that convergence will be achieved fastest by increasing the degree of the approximating polynomial. In a region where the solution is either not smooth or is not accurately approximated using a reasonably low-degree polynomial, refining the mesh at the locations of the nonsmoothness and using a polynomial of an appropriate (but relatively low) degree in each mesh interval would appear to be the best choice. Because the behavior of the solution to an arbitrary optimal control problem is not known a priori, it is useful to allow for both the polynomial degree to be different in each mesh interval and for the number and locations of the mesh intervals to be determined. Combining these last two features leads to a so-called hp method. In

this paper, we develop an hp method for solving optimal control problems using pseudospectral methods where higher-order polynomials are used in regions where the solution is smooth and larger numbers of low-degree polynomials are used in regions where the solution is not well approximated by polynomials of a reasonable degree.

V. Variable-Order *hp*-Adaptive Algorithm

A. Assessment of Approximation Error in a Mesh Interval

Consider again a K-interval Bolza optimal control problem, as defined in Sec. II, on the time interval $t \in [t_0, t_f]$. Let $[t_{k-1}, t_k]$ be the kth interval and suppose that the solution is collocated at N_k points on this interval. The goal of the hp-adaptive algorithm is to improve the accuracy of the solution in a computationally efficient manner by determining if a particular mesh interval in the current mesh has met a specified accuracy tolerance. If a mesh interval has not met the accuracy tolerance, then the number and distribution of collocation points needs to be modified either by increasing the degree of the approximating polynomial in the mesh interval and/or refining the mesh.

Let ϵ be an accuracy tolerance for the discretized differential-algebraic constraints (that is, the discretized dynamic and path constraints). The kth mesh interval is considered to be within the accuracy tolerance if the maximum violation in the differential-algebraic equations on $[t_{k-1}, t_k]$ is below ϵ. Unlike the state, which has a polynomial approximation, as given in Eq. (11), the control has no such defined approximation. While any interpolating function for which the value matches the approximation for the control at the N_k LGR collocation points in mesh interval k is acceptable, as far as the solution of the NLP is concerned, in this paper, the following N_kth-degree Lagrange polynomial is used to approximate the control in mesh intervals $1, \ldots, K-1$:

$$\mathbf{U}^{(k)}(\tau) = \sum_{i=1}^{N_k+1} \mathbf{U}_i^{(k)} \hat{\mathcal{L}}_i^{(k)}(\tau), \qquad \hat{\mathcal{L}}_j^{(k)}(\tau) = \prod_{\substack{i=1 \\ i \neq j}}^{N_k+1} \frac{\tau - \tau_i^{(k)}}{\tau_j - \tau_i^{(k)}} \tag{32}$$

Because the final time t_f is not collocated, the following $(N_k - 1)$th-degree Lagrange polynomial is used to approximate the control in the final mesh interval K:

$$\mathbf{U}^{(k)}(\tau) = \sum_{i=1}^{N_k} \mathbf{U}_i^{(k)} \tilde{\mathcal{L}}_i^{(k)}(\tau), \qquad \tilde{\mathcal{L}}_j^{(k)}(\tau) = \prod_{\substack{i=1 \\ i \neq j}}^{N_k} \frac{\tau - \tau_i^{(k)}}{\tau_j - \tau_i^{(k)}} \tag{33}$$

It is noted in Eq. (32) that, for $k \in [1, K-1]$, the support points of the Lagrange basis $\hat{\mathcal{L}}^{(k)}(\tau)$ are the N_k LGR points plus the noncollocated point $\tau = +1$, whereas in Eq. (33), the support points of the Lagrange basis $\tilde{\mathcal{L}}^{(k)}(\tau)$ are the N_k LGR points.

To estimate the error in the differential-algebraic equations, we insert the current state and control polynomials in the equations and evaluate them at L points $(\bar{\tau}_1^{(k)}, \ldots, \bar{\tau}_L^{(k)}) \in [t_{k-1}, t_k]$ in each mesh interval $k \in [1, \ldots, K]$:

$$|\dot{x}_i^{(k)}(\bar{\tau}_l^{(k)}) - f_i^{(k)}(\mathbf{X}_l^{(k)}, \mathbf{U}_l^{(k)}, \bar{\tau}_l^{(k)})| = a_{li}^{(k)} \tag{34}$$

$$\mathbf{C}_j^{(k)}(\mathbf{X}_l^{(k)}, \mathbf{U}_l^{(k)}, \bar{\tau}_l^{(k)}) = b_{lj}^{(k)} \tag{35}$$

where $1 \leq l \leq L$, $1 \leq i \leq n$, and $1 \leq j \leq s$. If every element of $a_{li}^{(k)}$ and $b_{lj}^{(k)}$ is less than ϵ on the current mesh interval, then the specified error tolerance ϵ has been satisfied on this interval. If any element of $a_{li}^{(k)}$ or $b_{lj}^{(k)}$ is greater than ϵ, then either the mesh interval will be refined or the degree of the polynomial on this interval will be increased.

It is noted that Eq. (35) measures the quality with which the path constraint of Eq. (3) is satisfied in mesh interval k at a point l. If we are in a region of the trajectory where the jth path constraint is inactive, $b_{lj}^{(k)}$ will be negative and, therefore, will always be less than ϵ. If we are in a region where the jth path constraint is active, $b_{lj}^{(k)}$ may have positive values. If any of these positive values exceeds ϵ, the path constraint is not satisfied to the desired accuracy tolerance.

Our error estimation process focuses on the error in the constraints. The accuracy in the first-order optimality conditions in the control problem is implicitly taken into account when we specify an error tolerance for the first-order optimality conditions in the NLP solver.

B. Increasing Polynomial Degree or Refining Mesh

If the accuracy in mesh interval k needs to be improved, the first step is to determine if the mesh should be refined or if the degree of the approximation should be increased. Suppose that $X_m^{(k)}(\tau)$ is the component of the state approximation in the kth mesh interval that corresponds to the maximum value of either $a_{li}^{(k)}$. The curvature of the mth component of the state in mesh interval k is given by

$$\kappa^{(k)}(\tau) = \frac{|\ddot{X}_m^{(k)}(\tau)|}{|[1 + \dot{X}_m^{(k)}(\tau)^2]^{3/2}|} \tag{36}$$

Let $\kappa_{\max}^{(k)}$ and $\bar{\kappa}^{(k)}$ be the maximum and mean value of $\kappa^{(k)}(\tau)$, respectively, where $\kappa^{(k)}(\tau)$ is computed using Eq. (36). Furthermore, let r_k be the ratio of the maximum to the mean curvature:

$$r_k = \frac{\kappa_{\max}^{(k)}}{\bar{\kappa}^{(k)}} \tag{37}$$

If $r_k < r_{\max}$, where $r_{\max} > 0$ is a user-defined parameter, the curvature is considered uniform in this mesh interval and a larger-degree polynomial is used to obtain a better approximation to mesh interval k. If $r_k > r_{\max}$, then there exists a large curvature relative to the rest of the mesh interval and the mesh is refined.

1. *Determination of New Polynomial Degree Within a Mesh Interval*

Suppose the result of Sec. V.B is that the polynomial degree within mesh interval k should be increased. Let M denote the initial degree of the polynomial in this interval. The degree N_k of the polynomial in mesh interval k is given by the formula

$$N_k = M + \mathbf{ceil}[\log_{10}(e_{\max}^{(k)}/\epsilon)] + A \tag{38}$$

where $A > 0$ is an arbitrary integer constant described in Sec. V.D, $e_{\max}^{(k)}$ is the maximum of $a_{li}^{(k)}$ and $b_{lj}^{(k)}$ over l, i, and j, and **ceil** is the operator that rounds to the next highest integer.

It is seen from Eq. (38) that the increase in the degree of the polynomial in the kth mesh interval is based on the ratio between the maximum error and the specified error tolerance. For a smooth function, a pseudospectral method exhibit exponential convergence. Hence, the growth in the polynomial degree is related to the log of the error. It is hoped that a unit increase in the polynomial degree will improve the error by one order of magnitude.

2. *Determination of Number and Placement of New Mesh Points*

Suppose the result of Sec. V.B is that the kth mesh interval should be refined. The following procedure is then used to determine the locations of the new mesh points. First, the new number of mesh intervals, denoted n_k, is computed as

$$n_k = \mathbf{ceil}[B\log_{10}(e_{\max}^{(k)}/\epsilon)] \tag{39}$$

where B is an arbitrary integer constant, described in Sec. V.D. The locations of the new mesh points are determined using the integral of a curvature density function in a manner similar to that given in [3]. Specifically, let $\rho(\tau)$ be the density function [3] given by

$$\rho(\tau) = c\kappa(\tau)^{1/3} \tag{40}$$

where c is a constant chosen, so that

$$\int_{-1}^{+1} \rho(\zeta)\, \mathrm{d}\zeta = 1$$

Let $F(\tau)$ be the cumulative distribution function given by

$$F(\tau) = \int_{-1}^{\tau} \rho(\zeta)\, \mathrm{d}\zeta \tag{41}$$

The n_k new mesh points are chosen so that

$$F(\tau_i) = \frac{i-1}{n_k}, \qquad 1 \le i \le n_k + 1$$

In [3], it is shown that distributing the mesh points in this way is, in some sense, optimal for the piecewise linear approximation of a curve when the error is measured in the L^1 norm. If $n_k = 1$, then no subintervals are created. Therefore, the minimum value for n_k should be at least two.

C. *hp*-Adaptive Algorithm

We now describe our hp-adaptive algorithm that uses the components described in Secs. V.A, V.B, V.B.1, and V.B.2. The algorithm starts by forming a coarse mesh, using polynomials of a fixed degree on each interval, and solving the resulting NLP. Based on the rules given next, for each interval in the mesh, we either refine the mesh or increase the degree of the polynomials. When we refine the mesh, we use a fixed degree M for the polynomials on each new mesh interval. When we increase the degree of the polynomials, we compute the new degree using the formula given in Eq. (38). We continue the refinement process until the specified error tolerance is satisfied. In more detail, the refinement process proceeds as follows:

1) Solve the NLP using the current mesh.

Begin: For $k = 1, \ldots, K$,

2) If $e_{\max}^{(k)} \le \epsilon$, then continue (proceed to next k).

3) If either $r_k \ge r_{\max}$ or $N_k > M$, then refine the kth mesh interval into n_k subintervals, where n_k is given by Eq. (39). Set the degree of the polynomials on each of the subintervals to be M and proceed to the next k.

4) Otherwise, set the degree of the polynomials on the kth subinterval to be N_k, which is given in Eq. (38).

End: **For** $k = 1, \ldots, K$.

5) Return to Step 1.

D. Remarks

In this algorithm, we alternate between a mesh refinement step in which the degree of the polynomials are set to M on each of the refined intervals and a step where we increase the degree of the polynomial according to Eq. (38). The goal of the hp-adaptive algorithm of this paper is to use as many low-degree mesh intervals as possible and to only use larger-degree polynomials in regions where the solution is smooth. Even for subdomains of the trajectory where larger-degree polynomials may be appropriate, the degree of the polynomial approximation is still small when compared with a p method. The reason for controlling the degree of the polynomial approximation is threefold. First, as discussed, the number of nonzero entries in the NLP constraint Jacobian grows as the square of the number of collocation points in each interval. Thus, if the degree of the approximating polynomial in a mesh interval becomes too large, the NLP becomes increasingly dense and the NLP solver has to work much harder to compute a solution. Second, the difference between the number of nonzero constraint Jacobian entries for an h method and hp method is small, even when the total number of collocation points is large, provided that the maximum polynomial degree in any mesh interval is small. Finally, due to the exponential convergence of a pseudospectral method in a mesh interval where the solution is smooth, it should be possible to obtain an accurate solution using a relatively small number of collocation points in that mesh interval. For all of the examples studied in this paper, a seventh-degree polynomial was the largest ever computed by the algorithm described in Sec. V.C.

Next, the algorithm uses the arbitrarily chosen parameters A, as described in Eq. (38) (where A controls the growth of the number of collocation points in a mesh interval) and B, as described in Eq. (39) (where B controls the growth in the number of mesh intervals). It is important to understand that a tradeoff exists between using large and small values of either A or B. If either A or B is sufficiently large, the algorithm will use fewer iterations to converge to an acceptable solution, but the size of the mesh or the number of collocation points may grow quickly between iterations. If either A or B is small, the mesh will grow much more slowly, but the algorithm may require many more iterations to achieve an acceptable solution.

The parameter $r_{\max}$ is used to determine whether to subdivide the current interval or increase the degree of approximating polynomial. For $r_{\max} < 1$, the strategy to refine the mesh will always occur and, simply, an h method will be used. If $r_{\max} = \infty$, then for each inaccurate mesh interval, the first refinement attempt will always be to increase the degree of approximating polynomial in the interval. By using a finite value of $r_{\max}$ greater than one, the algorithm will either subdivide the current mesh interval or increase the degree of polynomial approximation.

The parameter ϵ specifies the accuracy of the dynamic constraints and path constraints of the NLP approximation between collocation points. Specifically, the smaller the value of ϵ, the greater the required accuracy in the dynamics. In this study, we assume that the problem has been scaled so that ϵ is less than one. If the control problem is poorly scaled, then the corresponding NLP may not be solvable.

VI. Examples

The hp-adaptive method of Sec. V is now demonstrated on four examples. All examples were solved using a modified version of the open-source software GPOPS [16] with the NLP solver SNOPT [37], using a 2.5 GHZ Core 2 Duo Macbook Pro running Mac OS-X 10.5.8 with MATLAB R2009b. Note that algorithmic decisions depend on the choice of the NLP solver, and if a different solver was used that better exploits the sparsity of the discretized optimal control problem, then algorithmic decisions could change. All CPU times shown include only the time required to solve the NLP on all grids and do not include the setup time required to generate the grid for the subsequent mesh iteration. For all examples, the following values were chosen for the parameters described in Sec. V: $A = 1$, $B = 2$, $L = 30$, and $r_{\max} = 2$. To ensure that the NLP is solved to the same accuracy as the optimal control problem, the SNOPT feasibility and optimality tolerances are set to a value smaller than ϵ. Next, no upper limit was placed on the degree of the polynomial, as computed by Eq. (38), but a limit of $n_k \le 3B$ was placed on the growth of the mesh, as computed by Eq. (39). To obtain consistent results in all computations, the same stopping criterion is used for all the methods. In the particular case of a p method, if a solution is unacceptable, the degree of the global polynomial is increased arbitrarily by 10 until a solution is achieved. Furthermore, in order to ensure a fine sampling between the collocation points, a value $L = 1000$ is chosen for a p method. If a solution is not attained upon reaching a 200th-deg polynomial approximation, then it is decided that the p method was unable to obtain a solution. The parameter $r_{\max}$ controls the decision to either refine the mesh or increase the degree of the polynomial. For the h method, we set $r_{\max} < 1$ to force the algorithm to perform only mesh refinement (that is, the number of mesh intervals can change, but the degree of the polynomial in a mesh interval cannot change from its initial value). The terminology h-x denotes an h method of degree x in every mesh interval, while $hp - x$ denotes an hp method with a minimum polynomial degree of x in each interval. Finally, for examples 1 and 2, the maximum absolute error shown is the maximum absolute difference between the NLP solution and the exact solution at the state approximation points (that is, the LGR points on each mesh interval plus the terminal point).

A. Example 1

Consider again the motivating example given by Eqs. (19–21). It was seen in Sec. IV.A that the most rapid convergence was achieved using a p method. Suppose now that we consider the performance of the p, h-2, and hp-2 methods on this example. In the case of a p method, the initial mesh was two collocation points, while for the h method and the hp method, the initial mesh was 10 mesh intervals

Table 1 Summary of accuracy and speed for example 1, using various collocation strategies and accuracy tolerances

ϵ	E	Method	T, s	N_c	K	I	N_z
10^{-3}	2.40×10^{-7}	p	0.015	12	1	2	650
	5.96×10^{-3}	hp-2	0.0091	20	10	1	482
	5.96×10^{-3}	h-2	0.0091	20	10	1	482
10^{-4}	2.40×10^{-7}	p	0.015	12	1	2	650
	4.70×10^{-6}	hp-2	0.023	40	10	2	1,202
	6.66×10^{-4}	h-2	0.040	64	32	3	1,538
10^{-5}	2.40×10^{-7}	p	0.015	12	1	2	650
	1.21×10^{-7}	hp-2	0.030	50	10	2	1,652
	1.39×10^{-5}	h-2	0.089	160	80	3	3,842
10^{-6}	2.40×10^{-7}	p	0.015	12	1	2	650
	2.59×10^{-9}	hp-2	0.030	60	10	2	2,162
	5.22×10^{-7}	h-2	0.47	480	240	4	11,522
10^{-7}	2.40×10^{-7}	p	0.015	12	1	2	650
	4.87×10^{-11}	hp-2	0.037	70	10	2	2,732
	1.98×10^{-8}	h-2	4.63	1,568	784	5	37,634

with two collocation points per interval. Table 1 provides a summary of the results obtained for this example. When using the p method, the algorithm increased the polynomial degree once in order to meet the accuracy tolerance of all values of ϵ. In the case of the hp-2 method, it seen for $\epsilon = 10^{-3}$ that the algorithm did not iterate (that is, the solution on the first grid met the accuracy tolerance). Furthermore, for $\epsilon < 10^{-3}$, the mesh was never refined; only the degree of the polynomial was increased in order to satisfy the accuracy tolerance. Moreover, it is seen for $\epsilon = 10^{-3}$ and 10^{-4} that computational performance of the h-2 and hp-2 methods was similar. Finally, we observe for $\epsilon = 10^{-3}$ and 10^{-4} that the number of nonzero entries in the NLP constraint Jacobian using either the h-2 and hp-2 methods was essentially the same.

Suppose now that we analyze the computational performance of the different methods for $\epsilon < 10^{-3}$ down to $\epsilon = 10^{-7}$. First, it is seen that for all values of ϵ, from 10^{-3} to 10^{-7}, that computational efficiency using the p method and the hp-2 method are essentially the same, with the p method being slightly more efficient for the higher accuracy tolerances. Next, it is seen that, as ϵ is decreased from 10^{-5} to 10^{-7}, the hp-2 method is more computationally efficient than the h-2 method. We note again that, in the case of the hp-2 method, only the degree of the polynomial was increased in each mesh interval (that is, mesh refinement was never performed using the hp-2 method), and the accuracy tolerance was satisfied on the second grid iteration. The fact that mesh refinement was never employed using the hp-2 method is consistent with the fact that the solution to this problem is smooth. Interestingly, the maximum value of r_k on any mesh interval of the solution obtained on the initial grid was 1.32, well below the chosen value of $r_{\max} = 2$. By contrast, using the h-2 method with $\epsilon < 10^{-3}$, the algorithm chose to refine the mesh substantially. As a result, for higher accuracy tolerances, the h-2 method was less computationally efficient when compared with hp-2 method, while the p method showed minimal improvement in computational efficiency over the hp-2 method.

B. Example 2

Consider again the example given in Eqs. (23–26). Suppose we solve this problem using an hp-2 method with an initial mesh consisting of 10 uniform-width mesh intervals. Figures 3a–3d show the evolution of the control approximation on various meshes with $\epsilon = 10^{-3}$, where the interpolation is performed using Eqs. (32) and (33). On the first mesh, the discontinuity in the control is not captured accurately. On the subsequent meshes, however, more collocation points are placed near the location of the discontinuity, thus improving the accuracy with which the switch in the control is captured. On the initial grid, $r_k = 2.07$ for the interval containing the discontinuity, forcing subdivision of this interval. Figure 3e shows the difference between the time at which the maximum curvature occurs, denoted $t_\kappa^{\max}$, and the optimal switch time of the control t_s^* as a function of the iteration number. It is expected that, for an accurate solution, the location of maximum curvature in the state will occur at the same time as the discontinuity in the control. It is seen that, except for the second grid iteration, the maximum curvature in the state converges to the true location of the discontinuity in the control as the mesh is refined. Table 2 shows the performance of the algorithm using p, h-2, and hp-2 methods. First, it is interesting to note that the p method was never able to produce solutions for the given values of ϵ. When the p method was forced to stop at 200 collocation points, it resulted in a CPU time of 4.84 s and 82,002 nonzero Jacobian entries. In any case, the CPU time and number of nonzero NLP constraint Jacobian entries using the p method are significantly greater than those obtained using either the h-2 method or hp-2 method. Using either the h method or the hp-2 method with $\epsilon = 10^{-1}$, the accuracy tolerance was satisfied on the initial mesh. For $\epsilon = (10^{-2}, 10^{-3}, 10^{-4})$, the computational performance of the hp-2 and the h-2 methods was similar.

C. Example 3

Consider the following optimal control problem, which is a variation of the minimum time to climb of a supersonic aircraft [38–40]. Minimize the cost functional

$$J = t_f \tag{42}$$

subject to the dynamic constraints

$$\dot{h} = v \sin\gamma, \qquad \dot{v} = \frac{F_t - F_d}{m} - F_g \sin\gamma, \qquad \dot{\gamma} = \frac{F_g}{v}(\eta - \cos\gamma) \tag{43}$$

the boundary conditions

$$h(0) = 0 \text{ m}, \qquad h(t_f) = 19995 \text{ m}, \qquad v(0) = 129.31 \text{ m/s}$$
$$v(t_f) = 295.09 \text{ m/s}, \qquad \gamma(0) = 0 \text{ deg}, \qquad \gamma(t_f) = 0 \text{ deg} \tag{44}$$

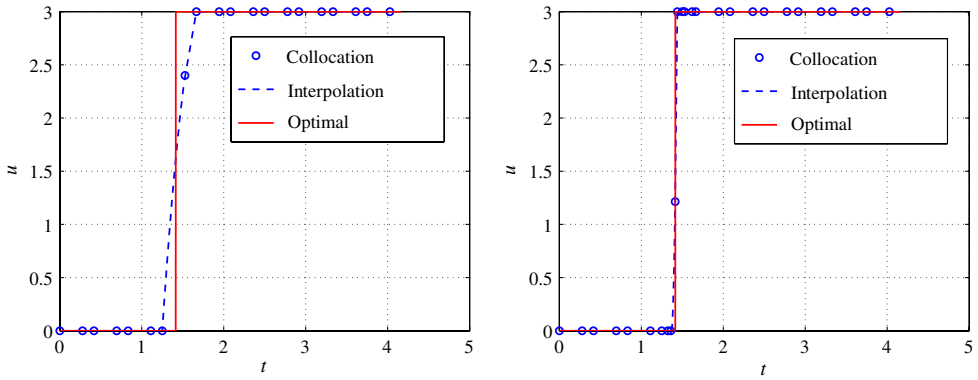
and the inequality path constraints

$$h \geq 0, \qquad \gamma \leq \gamma_{\max} \tag{45}$$

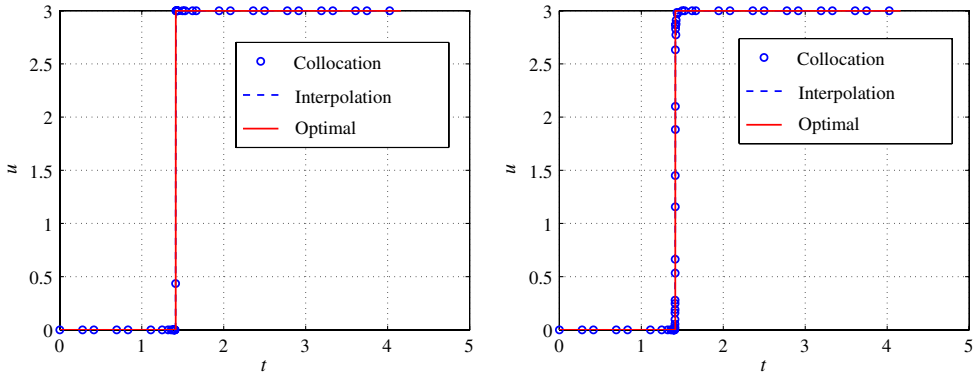
To force the flight-path angle path constraint $\gamma \leq \gamma_{\max}$ to be active on the optimal solution, we choose $\gamma_{\max} = 45$ deg. Additional details for the vehicle used in this problem can be found in [39,40].

This example was solved using the p, h-x, and hp-x methods, where the h-x and hp-x methods were initialized with a mesh consisting of 10 uniform mesh intervals with x collocation points in each mesh interval and the p solutions were initialized using 20 collocation points on the time interval. Figures 4a–4d show the flight-path angle on iterations (1, 2, 4, and 8) for the hp-4 method using $\epsilon = 10^{-3}$. It is seen that the initial mesh does not accurately approximate the optimal solution in either the takeoff region (from $t = 0$ to $t \approx 10$ s) or in the region near path constraint (that is, from $t \approx 20$ s to $t \approx 35$ s). As the mesh refinement progresses, however, it is seen that the solution in these two regions improves dramatically, leading to much higher accuracy on the final mesh where the majority of mesh points and collocation points are placed in the region from $t = 0$ to $t \approx 40$ s. Finally, in the region $t \in [\approx 50, t_f]$, it is observed that the mesh does not change from the second mesh iteration onwards.

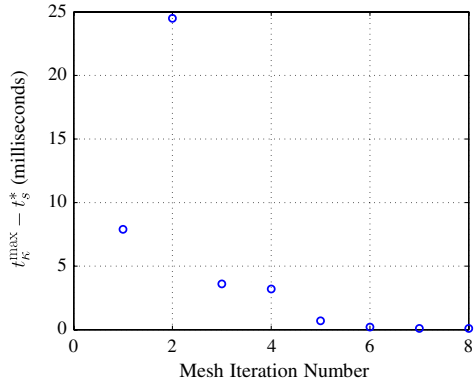
Table 3 shows the computation times and number of nonzero NLP constraint Jacobian entries for this example using the p, h-2, and hp-x methods. It is seen that a solution is obtained using a p method for $\epsilon = (10^{-1}, 10^{-2}, 10^{-3})$. No solution was obtained for $\epsilon = 10^{-4}$, as the p method was stopped at 200 collocation points. Also, using either the h-2 method or the hp-x methods, it is seen that the initial mesh satisfies the accuracy tolerance for $\epsilon = 10^{-1}$. For $\epsilon = (10^{-1}, 10^{-2})$, the performance of the h-2, hp-x, and p methods are similar. For $\epsilon = 10^{-3}$, it is seen that the p method is significantly less computationally efficient, while the h-2 and hp-x methods have similar computational performance. No p-method solution was found for $\epsilon = 10^{-4}$. The resulting CPU time and nonzero Jacobian entries were 1100.4 s and 123,602, respectively. When $\epsilon = 10^{-4}$, a



a) Mesh 1 b) Mesh 2



c) Mesh 4 d) Mesh 8



e) Difference between the time at which maximum curvature Occurs, $t_K^{\max}$, and the optimal switch time in the control, t_s^*, as a function of mesh iteration number

Fig. 3 Control for example 2 on meshes a) 1, b) 2, c) 4, and d) 8, using an *hp*-2 method with an accuracy tolerance of $\epsilon = 10^{-3}$, and e) $t_\kappa^{\max} - t_s^*$ vs mesh iteration number.

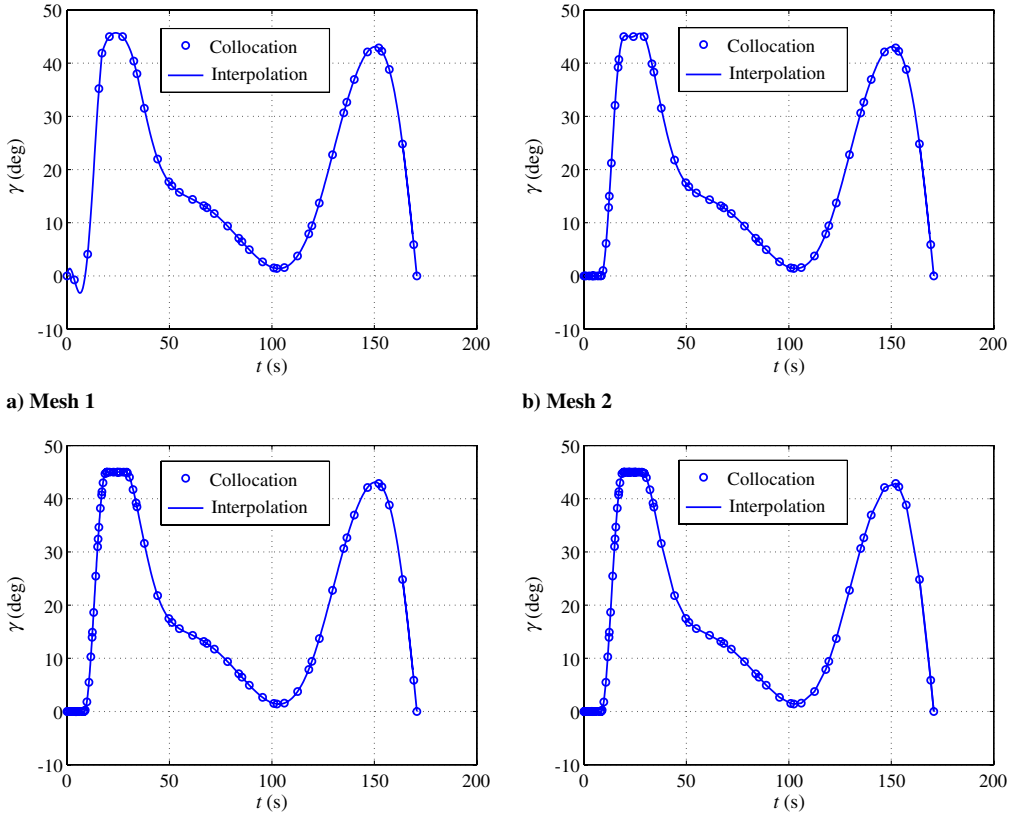
significant improvement is seen using higher-order hp-x methods, as compared with the h-2 method. In comparing the h-2 and hp-2 methods, it is important to observe that allowing the flexibility to increase the degree of the polynomial from its minimum allowable value greatly reduces the total number of collocation points. Furthermore, the hp-4 method showed the best performance for $\epsilon = 10^{-4}$. Thus, while the hp-4 method results in the denser NLP, it also turns out that the hp-4 method has both the fewest number of collocation points and the fewest number of nonzero Jacobian entries. This example again demonstrates the relative ineffectiveness of the p method over either the h-2 method or an hp-x method when an active path constraint is present.

D. Example 4

Consider the following optimal control problem [1] of maximizing the crossrange during the atmospheric entry of a reusable launch vehicle. Minimize the cost functional

Table 2 Summary of accuracy and speed for example 2, using various collocation strategies and accuracy tolerance

ϵ	E	Method	T, s	N_c	K	I	N_z
10^{-1}	——	p	——	——	——	——	——
	2.74×10^{-2}	hp-2	0.097	20	10	1	282
	2.74×10^{-2}	h-2	0.097	20	10	1	282
10^{-2}	——	p	——	——	——	——	——
	4.96×10^{-3}	hp-2	0.12	40	20	8	562
	7.01×10^{-3}	h-2	0.10	28	14	5	394
10^{-3}	——	p	——	——	——	——	——
	3.20×10^{-3}	hp-2	0.13	50	24	8	718
	6.32×10^{-3}	h-2	0.096	36	18	5	506
10^{-4}	——	p	——	——	——	——	——
	6.97×10^{-4}	hp-2	0.29	86	42	11	1222
	7.20×10^{-4}	h-2	0.20	72	36	8	1010



a) Mesh 1 b) Mesh 2

c) Mesh 4 d) Mesh 8

Fig. 4 Flight-path angle γ (deg) vs t for example 3 on mesh iterations 1, 2, 4, and 8, using the hp-4 method with an accuracy tolerance of $\epsilon = 10^{-3}$.

$$J = -\phi(t_f) \tag{46}$$

subject to the dynamic constraints

$$\begin{aligned}
&\dot{r} = v\sin\gamma, \qquad \dot{\theta} = \frac{v\cos\gamma\sin\psi}{r\cos\phi}, \qquad \dot{\phi} = \frac{v\cos\gamma\cos\psi}{r}\\
&\dot{v} = -\frac{F_d}{m} - F_g\sin\gamma, \qquad \dot{\gamma} = \frac{F_l\cos\sigma}{mv} - \left(\frac{F_g}{v} - \frac{v}{r}\right)\cos\gamma\\
&\dot{\psi} = \frac{F_l\sin\sigma}{mv\cos\gamma} + \frac{v\cos\gamma\sin\psi\tan\phi}{r}
\end{aligned} \tag{47}$$

and the boundary conditions

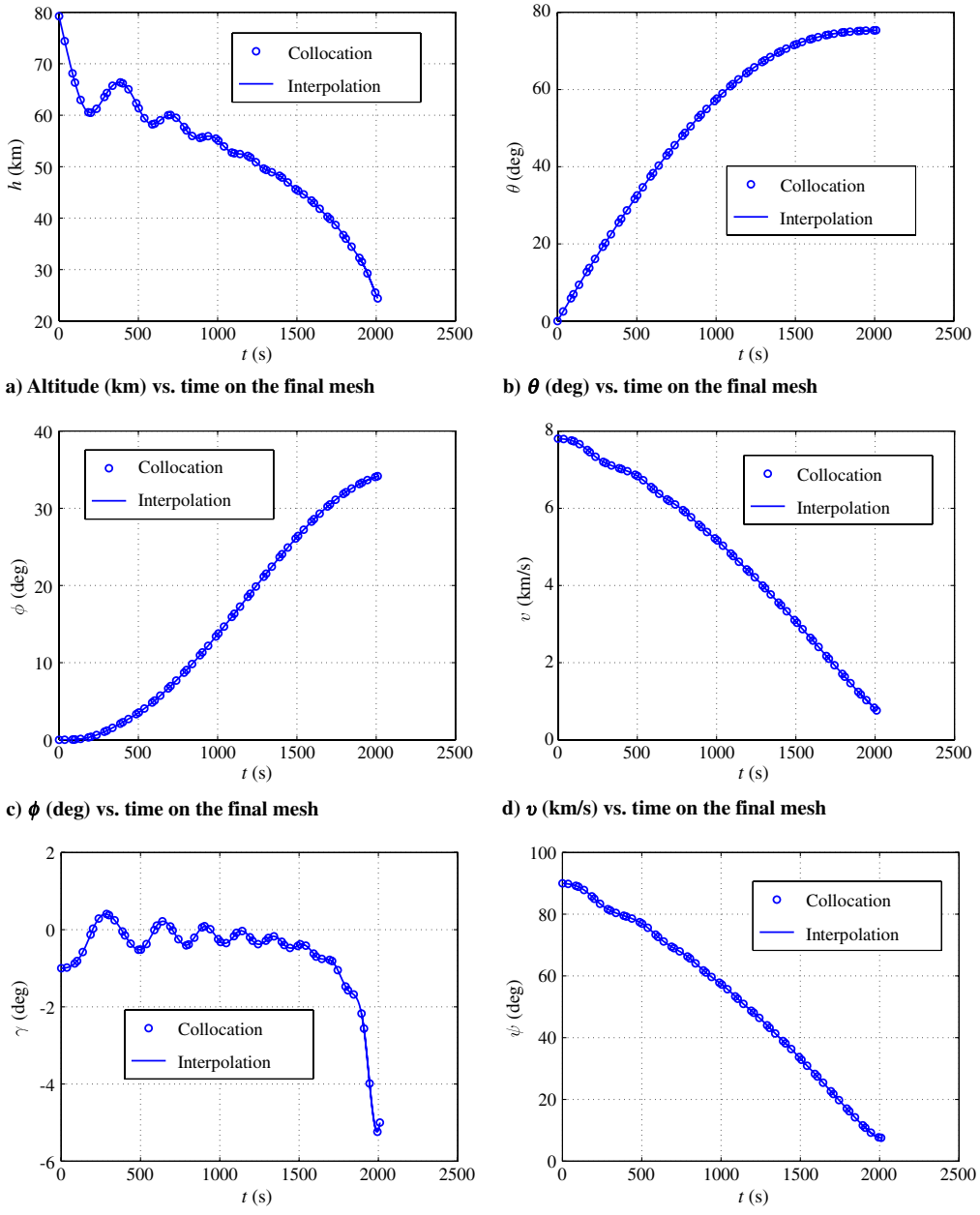
$$\begin{aligned}
&r(0) = 79248 + R_e \text{ m}, \qquad r(t_f) = 24384 + R_e \text{ m}\\
&\theta(0) = 0 \text{ deg}, \qquad \theta(t_f) = \text{free}, \qquad \phi(0) = 0 \text{ deg}\\
&\phi(t_f) = \text{free}, \qquad v(0) = 7803 \text{ m/s}, \qquad v(t_f) = 762 \text{ m/s}\\
&\gamma(0) = -1 \text{ deg}, \qquad \gamma(t_f) = -5 \text{ deg}, \qquad \psi(0) = 90 \text{ deg}\\
&\psi(t_f) = \text{free}
\end{aligned} \tag{48}$$

Further details of this problem, including the aerodynamic model, can be found in [1]. It is noted that, unlike either of the previous two examples where either the control was discontinuous or an inequality path constraint was active, this problem has a much smoother solution.

This example was solved using the p, h-x, and hp-x methods, where the h-x and hp-x methods were initialized with an initial mesh consisting of 20 uniform mesh intervals with x collocation points in each mesh interval and the p solutions were initialized using 20 collocation points on the time interval. Figures 5a–5f show the state obtained using the hp algorithm of Sec. V, where the interpolation is performed using Eq. (11). Table 4 shows the computational performance using the p, h-x, and hp-x methods. Because the optimal solution to this example is smooth, the p method was successful for all values of ϵ. Furthermore, for $\epsilon = (10^{-4}, 10^{-5})$, the solutions using any of the methods were essentially the same. As the accuracy tolerance is tightened, however, the h-2 method requires a very large number of collocation points to meet the accuracy tolerance. As a result, a significantly larger computation time is required to solve the NLP using the h-2 method when compared with using the hp-x methods. In addition, while the p method resulted in solutions with the fewest number of collocation points, the NLP that results from the use of high-degree polynomials is much more dense than the NLPs arising from the hp-x and h methods. This increased NLP density in turn increases the computational time required to solve the problem. In particular, it is seen for $\epsilon = 10^{-6}$ that the hp-x

Table 3 Summary of accuracy and speed for example 3, using various collocation strategies and accuracy tolerances

ϵ	Method	T, s	N_c	K	I	N_z
10^{-1}	p	0.20	30	1	2	3,424
	hp-4	0.16	40	10	1	1,202
	hp-3	0.083	30	10	1	812
	hp-2	0.064	20	10	1	482
	h-2	0.064	20	10	1	482
10^{-2}	p	0.37	40	1	3	5,522
	hp-4	0.32	48	12	3	1,442
	hp-3	0.22	35	11	3	977
	hp-2	0.51	38	17	10	962
	h-2	0.42	46	23	7	1,106
10^{-3}	p	21.45	110	1	10	38,282
	hp-4	1.59	95	23	8	2,903
	hp-3	2.38	118	38	14	3,248
	hp-2	3.03	117	43	14	3,191
	h-2	1.35	138	69	7	3,314
10^{-4}	p	———	———	———	———	———
	hp-4	7.33	192	46	10	5,906
	hp-3	8.09	234	72	10	6,608
	hp-2	29.14	245	95	20	6,617
	h-2	18.88	334	167	13	8,018



a) Altitude (km) vs. time on the final mesh

b) θ (deg) vs. time on the final mesh

c) ϕ (deg) vs. time on the final mesh

d) v (km/s) vs. time on the final mesh

e) γ (deg) vs. time on the final mesh

f) ψ (deg) vs. time on the final mesh

Fig. 5 State on the final mesh for example 4, using the *hp*-3 method with an accuracy tolerance of $\epsilon = 10^{-5}$.

methods produce solutions using between approximately one-third to one-half the computation time required by the p method. For $\epsilon = 10^{-7}$, the hp-3 and hp-4 methods significantly outperform the h-2 method and are slightly more computationally efficient than the p method. As a result, while the hp-3 and hp-4 methods use more than double the number of collocation points used by the p method, the increased sparsity of the hp-3 and hp-4 methods lead to a more computationally efficient NLP. Furthermore, the hp-3 and hp-4 methods use many fewer collocation points than are used by the h-2 method, again resulting in increased computational efficiency. As the accuracy requirements are increased on this problem, an hp method outperforms an h method.

VII. Discussion of Results

The application of the hp-adaptive algorithm to the four examples highlights several key aspects of using the hp-adaptive approach. In some cases, a strategy biased toward a p method works best, while in others, a strategy biased toward an h method is better. In general, it will not be possible to accurately capture discontinuities or rapid changes in the trajectory using a p method, while an h method is less able to exploit smoothness in the solution. By using an hp method, it is possible to efficiently determine accurate solutions to problems that may have a variety of different behaviors in different regions of the solution. In general, it is preferable to start with low-degree polynomial approximations in each mesh interval, because the meshes are generated much more quickly than would be the case had a high-degree p method been used for the initial mesh; after a solution has been obtained on the first mesh, the solutions on the subsequent meshes are obtained very quickly. In addition, it is seen that the hp-adaptive method greatly outperforms the p method in the cases of a discontinuity in the control or an active path constraint. For smoother problems, the hp-adaptive method performs similarly or better to the p method due to increased computational sparsity in the NLP. Comparing the hp method to an h method, it is seen for low-accuracy tolerances that the hp method is at least as fast, if not faster, than an h method. When it is required to satisfy a high-accuracy tolerance, it was found that the speed improvement of the hp method over the h method can be quite significant. This last result indicates that allowing for a moderate increase in the degree of the polynomial

Table 4 Summary of accuracy and speed for example 4, using various collocation strategies and accuracy tolerances

ϵ	Method	T, s	N_c	K	I	N_z
10^{-4}	p	7.30	40	1	3	12,002
	hp-4	3.61	80	20	1	6,722
	hp-3	3.33	60	20	1	4,682
	hp-2	1.68	45	20	2	3,380
	h-2	2.2	50	25	3	3,602
10^{-5}	p	7.30	40	1	3	12,002
	hp-4	5.74	88	22	2	7,394
	hp-3	7.97	71	23	5	5,600
	hp-2	6.44	71	20	5	5,934
	h-2	6.39	92	46	4	6,626
10^{-6}	p	28.79	70	1	6	33,602
	hp-4	10.88	124	28	2	10,850
	hp-3	13.16	129	37	4	10,604
	hp-2	13.82	119	30	5	10,232
	h-2	47.32	334	167	5	24,050
10^{-7}	p	41.68	80	1	7	43,202
	hp-4	32.20	196	39	3	18,014
	hp-3	27.07	192	50	4	16,430
	hp-2	233.30	387	88	9	33,764
	h-2	1,247.40	1030	515	10	74,162

in a mesh interval can actually reduce execution time, because a solution can be obtained using a much smaller NLP than may be required using an h method.

VIII. Conclusions

A variable-order adaptive control algorithm has been developed for solving optimal control problems using pseudospectral methods. In the method of this paper, accuracy is improved either by refining the mesh or increasing the degree of the polynomial approximation in particular mesh intervals. The mesh refinement is based on the integral of the curvature. The adaptive scheme is demonstrated on four examples, including a problem for which the solution is nonsmooth and a problem with an active inequality path constraint. When compared with a global pseudospectral method or a fixed-degree method, it is found that the approach of this paper was more efficient computationally, while the computed solutions were of comparable accuracy. For problems with a smooth solution, the adaptive method converged by increasing the degree of the polynomial, while for problems where the solution was nonsmooth, the adaptive method refined the mesh in the nonsmooth part of the domain.

Acknowledgments

The authors gratefully acknowledge support for this research from the U. S. Air Force under contract FA8651-08-D-0108, from the NASA Florida Space Grant Consortium under grant NNG05GK00H, from the National Science Foundation under grant 0620286, and from the U.S. Office of Naval Research under grant N00014-11-1-0068.

References

[1] Betts, J. T., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed., SIAM Press, Philadelphia, 2009, pp. 152–166.

[2] Jain, D., and Tsiotras, P., "Trajectory Optimization Using Multiresolution Techniques," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 5, Sept.–Oct. 2008, pp. 1424–1436. doi:10.2514/1.32220

[3] Zhao, Y., and Tsiotras, P., "A Density-Function Based Mesh Refinement Algorithm for Solving Optimal Control Problems," Proceedings of the Infotech and Aerospace Conference, Seattle, WA, AIAA Paper 2009-2019, April 2009.

[4] Betts, J. T., and Huffman, W. P., "Mesh Refinement in Direct Transcription Methods for Optimal Control," *Optimal Control Applications and Methods*, Vol. 19, No. 1, 1998, pp. 1–21. doi:10.1002/(SICI)1099-1514(199801/02)19:1<1::AID-OCA616>3.0.CO;2-Q

[5] Cuthrell, J. E., and Biegler, L. T., "On the Optimization of Differential-Algebraic Processes," *AIChE Journal*, Vol. 33, No. 8, Aug. 1987, pp. 1257–1270. doi:10.1002/aic.690330804

[6] Cuthrell, J. E., and Biegler, L. T., "Simultaneous Optimization and Solution Methods for Batch Reactor Control Profiles," *Computers and Chemical Engineering*, Vol. 13, Nos. 1–2, 1989, pp. 49–62. doi:10.1016/0098-1354(89)89006-4

[7] Elnagar, G., Kazemi, M., and Razzaghi, M., "The Pseudospectral Legendre Method for Discretizing Optimal Control Problems," *IEEE Transactions on Automatic Control*, Vol. 40, No. 10, 1995, pp. 1793–1796. doi:10.1109/9.467672

[8] Elnagar, G., and Razzaghi, M., "Short Communication: A Collocation-Type Method for Linear Quadratic Optimal Control Problems," *Optimal Control Applications and Methods*, Vol. 18, No. 3, 1997, pp. 227–235. doi:10.1002/(SICI)1099-1514(199705/06)18:3<227::AID-OCA598>3.0.CO;2-A

[9] Fahroo, F., and Ross, I. M., "Costate Estimation by a Legendre Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, 2001, pp. 270–277. doi:10.2514/2.4709

[10] Fahroo, F., and Ross, I. M., "Direct Trajectory Optimization by a Chebyshev Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 160–166. doi:10.2514/2.4862

[11] Fahroo, F., and Ross, I. M., "Pseudospectral Methods for Infinite-Horizon Nonlinear Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 4, 2008, pp. 927–936. doi:10.2514/1.33117

[12] Benson, D. A., "A Gauss Pseudospectral Transcription for Optimal Control," Ph.D. Thesis, Department of Aeronautics and Astronautics, Massachusetts Inst. of Technology, Cambridge, MA, Nov. 2004.

[13] Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Rao, A. V., "Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 6, Nov.–Dec. 2006, pp. 1435–1440. doi:10.2514/1.20478

[14] Huntington, G. T., "Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, May 2007.

[15] Rao, A. V., Benson, D. A., Darby, C. L., Francolin, C., Patterson, M. A., Sanders, I., and Huntington, G. T., "Algorithm 902: GPOPS, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method," *ACM Transactions on Mathematical Software*, Vol. 37, No. 2, April–June 2010, Paper 22. doi:10.1145/1731022.1731032

[16] Rao, A. V., Benson, D. A., Darby, C. L., Patterson, M. A., Sanders, I., and Huntington, G. T., "User's Manual for GPOPS Version 3.3: A MATLAB Software for Solving Optimal Control Problems Using Pseudospectral Methods," , Dec. 2010, http://www.gpops.org/gpopsManual.pdf [retrieved 3 Jan. 2011].

[17] Kameswaran, S., and Biegler, L. T., "Convergence Rates for Direct Transcription of Optimal Control Problems Using Collocation at Radau Points," *Computational Optimization and Applications*, Vol. 41, No. 1, 2008, pp. 81–126. doi:10.1007/s10589-007-9098-9

[18] Garg, D., Patterson, M. A., Darby, C. L., Francolin, C., Huntington, G. T., Hager, W. W., and Rao, A. V., "Direct Trajectory Optimization and Costate Estimation of Finite-Horizon and Infinite-Horizon Optimal Control Problems via a Radau Pseudospectral Method" [online], *Computational Optimization and Applications*, Oct. 2009, http://www.springerlink.com/content/n851q6n343p9k60k/ [retrieved Jan. 2011]. doi:10.1007/s10589-009-9291-0

[19] Garg, D., Patterson, M. A., Hager, W. W., Rao, A. V., Benson, D. A., and Huntington, G. T., "A Unified Framework for the Numerical Solution of Optimal Control Problems Using Pseudospectral Methods," *Automatica*, Vol. 46, No. 11, Nov. 2010, pp. 1843–1851. doi:10.1016/j.automatica.2010.06.048

[20] Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics*, Spinger–Verlag, Heidelberg, Germany, 1988.

[21] Fornberg, B., *A Practical Guide to Pseudospectral Methods*, Cambridge Univ. Press, New York, 1998.

[22] Trefethen, L. N., *Spectral Methods Using MATLAB*, SIAM Press, Philadelphia, 2000.

[23] Babuska, I., and Suri, M., "The p and hp Version of the Finite Element

Method, an Overview," *Computer Methods in Applied Mechanics and Engineering*, Vol. 80, Nos. 1–3, 1990, pp. 5–26. doi:10.1016/0045-7825(90)90011-A

[24] Babuska, I., and Suri, M., "The p and hp Version of the Finite Element Method, Basic Principles and Properties," *SIAM Review*, Vol. 36, No. 4, 1994, pp. 578–632. doi:10.1137/1036141

[25] Gui, W., and Babuska, I., "The h, p, and hp Versions of the Finite Element Method in 1 Dimension. Part I. The Error Analysis of the p Version," *Numerische Mathematik*, Vol. 49, No. 6, 1986, pp. 577–612. doi:10.1007/BF01389733

[26] Gui, W., and Babuska, I., "The h, p, and hp Versions of the Finite Element Method in 1 Dimension. Part II. The Error Analysis of the h and h-p Versions," *Numerische Mathematik*, Vol. 49, No. 6, 1986, pp. 613–657. doi:10.1007/BF01389734

[27] Gui, W., and Babuska, I., "The h, p, and hp Versions of the Finite Element Method in 1 Dimension. Part III. The Adaptive h-p Version," *Numerische Mathematik*, Vol. 49, No. 6, 1986, pp. 659–683. doi:10.1007/BF01389735

[28] Galvao, A., Gerritsma, M., and Maerschalk, B. D., "hp-Adaptive Least-Squares Spectral Element Method for Hyperbolic Partial Differential Equations," *Journal of Computational and Applied Mathematics*, Vol. 215, No. 2, June 2008, pp. 409–418. doi:10.1016/j.cam.2006.03.063

[29] Heinrichs, W., "An Adaptive Least Squares Scheme for the Burgers Equation," *Numerical Algorithms*, Vol. 44, No. 1, January 2007, pp. 1–10. doi:10.1007/s11075-007-9071-9

[30] Dorao, C. A., and Jakobsen, H. A., "hp-Adaptive Least Squares Spectral Element Method for Population Balance Equations," *Applied Numerical Mathematics*, Vol. 58, No. 5, May 2008, pp. 563–576. doi:10.1016/j.apnum.2006.12.005

[31] Dorao, C. A., Fernandino, M., Jakobsen, H. A., and Svendsen, H. F., "hp-Adaptive Spectral Element Solver for Reactor Modeling," *Chemical Engineering Science*, Vol. 64, No. 5, March 2009, pp. 904–911. doi:10.1016/j.ces.2008.10.027

[32] Karniadakis, G., and Sherwin, S., *Spectral/hp Element Methods for CFD*, Oxford Univ. Press, Oxford, England, U.K., 1999.

[33] Darby, C. L., Hager, W. W., and Rao, A. V., "An hp-Adaptive Pseudospectral Method for Solving Optimal Control Problems," *Optimal Control Applications and Methods* [online journal], Aug. 2010, http://onlinelibrary.wiley.com/doi/10.1002/oca.957/abstract [retrieved 2010]. doi:10.1002/oca.957

[34] Abramowitz, M., and Stegun, I., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, New York, 1965, pp. 916–918.

[35] Soueres, P., and Boissonnat, J. D., "Robot Motion Planning and Control," Springer, Englewood Cliffs, NJ, 1998, p. 95.

[36] Meditch, J., "On the Problem of Optimal Thrust Programming for a Soft Lunar Landing," *IEEE Transactions on Automatic Control*, Vol. 9, No. 4, 1964, pp. 477–484. doi:10.1109/TAC.1964.1105758

[37] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131. doi:10.1137/S0036144504446096

[38] Bryson, A. E., Desai, M. N., and Hoffman, W. C., "Energy-State Approximation in Performance Optimization of Supersonic Aircraft," *Journal of Aircraft*, Vol. 6, No. 6, 1969, pp. 481–488. doi:10.2514/3.44093

[39] Seywald, H., and Kumar, R. R., "Finite Difference Scheme for Automatic Costate Calculation," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 1, 1996, pp. 231–239. doi:10.2514/3.21603

[40] Rao, A. V., "Application of a Dichotomic Basis Method to Performance Optimization of Supersonic Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 3, May–June 2000, pp. 570–573. doi:10.2514/2.4570

D. Spencer
Associate Editor