JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 42, No. 9, September 2019

Computational Method for Optimal Guidance and Control Using Adaptive Gaussian Quadrature Collocation

Miriam E. Dennis,* William W. Hager,† and Anil V. Rao‡
University of Florida, Gainesville, Florida 32611

DOI: 10.2514/1.G003943

A method is described for computational optimal guidance and control using adaptive Gaussian quadrature collocation and sparse nonlinear programming. The method employs adaptive Legendre–Gauss–Radau (LGR) quadrature collocation using a mesh truncation and remapping procedure at the start of each guidance cycle, thereby retaining only the mesh points associated with the unexpired horizon. Using this approach, the starting mesh for a guidance cycle is smaller than the starting mesh used on the previous guidance cycle. The nonlinear programming problem is then solved on the reduced mesh corresponding to the unexpired horizon to generate the control that is used on the current guidance cycle. It is noted that the mesh points on the unexpired horizon are well placed for rapidly solving the LGR quadrature collocation approximation of the optimal control problem. The method developed in this paper is demonstrated on two challenging aerospace optimal control problems. For both examples, the dynamics are simulated on each guidance cycle using a perturbed dynamic model in the absence and presence of a computation time delay. The results of this study demonstrate that the method developed in this paper is viable as a computational method for optimal guidance and control.

Nomenclature

$\boldsymbol{a}$	=	vector field that defines reference dynamics
$\tilde{\boldsymbol{a}}$	=	vector field that defines perturbed dynamics
$\boldsymbol{b}_f$	=	vector field that defines terminal conditions
$\boldsymbol{b}_0$	=	vector field that defines initial conditions
C	=	computation time
$\boldsymbol{c}$	=	vector field that defines path constraints
$E^{(s)}$	=	number of mesh intervals in expired horizon on guidance cycle s
G	=	guidance cycle duration
$\mathcal{I}_k$	=	mesh interval k on $[-1, +1]$
K	=	total number of mesh intervals
k	=	mesh interval number
$\mathcal{L}$	=	Lagrangian of Bolza cost functional
$\mathcal{M}$	=	endpoint cost of Bolza cost functional
M	=	total number of mesh refinement iterations
N	=	total number of Legendre–Gauss–Radau quadrature collocation points on a mesh
N_k	=	number of Legendre–Gauss–Radau quadrature collocation points in mesh interval k
S	=	total number of guidance cycles
s	=	guidance cycle
T	=	mesh time
T_d	=	time delay
$T_e^{(s)}$	=	mesh time on $[-1, +1]$ corresponding to terminal time on expired horizon on guidance cycle s
T_k	=	kth mesh point on mesh time $T \in [-1, +1]$
t	=	time on $[t_0, t_f]$
$t_e^{(s)}$	=	time on $[t_0^{(s)}, t_f^{(s)}]$ corresponding to terminal time on expired horizon on guidance cycle s
t_f	=	terminal time
t_0	=	initial time
$\boldsymbol{u}$	=	control input to dynamical system
$\boldsymbol{u}^*$	=	optimal control
$\boldsymbol{Y}$	=	Lagrange polynomial approximation of state of dynamical system
$\boldsymbol{Y}_j^{(k)}$	=	approximation of state of dynamical system at discretization point j in mesh interval k
$\boldsymbol{y}$	=	state of dynamical system
$\boldsymbol{y}^*$	=	optimal state of dynamical system
$\tilde{\boldsymbol{y}}$	=	state of dynamical system using perturbed dynamics
$\Delta \boldsymbol{y}(t)$	=	$\tilde{\boldsymbol{y}}(t) - \boldsymbol{y}^*(t)$
τ	=	time on $[-1, +1]$

I. Introduction

OPTIMAL control problems arise frequently in many engineering applications due to the need to optimize performance of a controlled dynamical system. In general, optimal control problems do not have analytic solutions and, thus, must be solved numerically. Numerical methods for optimal control fall into two broad categories: indirect methods and direct methods. In an indirect method, the first-order variational optimality conditions are derived, and the optimal control problem is converted to a Hamiltonian boundary-value problem (HBVP). The HBVP is then solved numerically using a differential-algebraic equation solver. In a direct method, the state and control are approximated, and the optimal control problem is transcribed into a finite-dimensional nonlinear programming problem (NLP) [1]. The NLP is then solved numerically using well-developed software [2–4]. Even in cases where an accurate approximation to the solution of an optimal control problem can be computed, disturbances in the actual system along with measurement errors lead to suboptimal performance and constraint violations in the actual system. If the difference between the motion of the reference system and that of the actual system is large, it may be necessary to re-solve the optimal control problem (i.e., perform an optimal midcourse correction) in real-time based on the current state of the actual system. In cases where it is necessary to re-solve the optimal control problem, it is desirable that the new optimal solution be obtained sufficiently quickly so that this new solution can be implemented in the actual system. Because most optimal control problems must be solved numerically, in order to realize real-time optimal control in a constrained nonlinear dynamical system, it is necessary to develop computational methods.

Preliminary results presented as Paper 2018-0848 at the 2018 AIAA Guidance, Navigation, and Control Conference, Kissimmee, FL, 8–12 January 2018; received 3 July 2018; revision received 7 March 2019; accepted for publication 8 March 2019; published online Open Access 3 May 2019.

*Ph.D. Student, Department of Mechanical and Aerospace Engineering; dennism09@ufl.edu.

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

‡Associate Professor, University Term Professor, and Erich Farber Faculty Fellow, Department of Mechanical and Aerospace Engineering; anilvrao@ufl.edu. Associate Fellow AIAA (Corresponding Author).

Over the past two decades, a particular class of numerical methods, called direct collocation methods, has been used extensively for solving continuous optimal control problems. A direct collocation method is an implicit simulation method where the state and control are parameterized, and the constraints in the continuous optimal control problem are enforced at a specially chosen set of collocation points. Traditional direct collocation methods take the form of an h method (e.g., Euler or Runge–Kutta methods) where the domain of interest is divided into a mesh, the state is approximated using the same fixed-degree polynomial in each mesh interval, and convergence is achieved by increasing the number and placement of the mesh points [1]. In contrast to an h method, in recent years so-called p methods have been developed. In a p method, the number of intervals is fixed, and convergence is achieved by increasing the degree of the approximation in each interval. To achieve maximum effectiveness, p methods have been developed using *orthogonal collocation at Gaussian quadrature points* [5–12]. For problems whose solutions are smooth and well-behaved, a Gaussian quadrature orthogonal collocation method converges at an exponential rate [9–11]. Gauss quadrature collocation methods use either Legendre–Gauss (LG) points [5–7,10], Legendre–Gauss–Radau (LGR) points [6–9,11], or Legendre–Gauss–Lobatto (LGL) points [12].

Although h methods have been used extensively and p methods are useful on particular types of problems, both h and p approximations have limitations. In the case of an h method, it may be required to use an extremely fine mesh to improve accuracy. In the case of a p method, it may be required to use an unreasonably large degree polynomial to improve accuracy. To reduce significantly the size of the finite-dimensional approximation, and thus improve computational efficiency of solving the NLP, in recent years a new class of hp collocation methods has been developed for solving optimal control problems. The class of hp methods was originally developed for solving partial differential equations [13–17]. The extension of these techniques into an optimal control setting is not straightforward because for an optimal control problem it is not only necessary to solve a differential equation, but it is also necessary to solve an optimization problem in order to determine an approximation of the optimal control. The difference between the numerical solution of a differential equation and the numerical solution of an optimal control problem can be seen using a Runge–Kutta method. When solving a differential equation, a seventh-order Runge–Kutta method must satisfy 48 conditions; on the other hand, when solving an optimal control problem, a corresponding seventh-order Runge–Kutta scheme must satisfy 4116 conditions [18,19]. To overcome the intractability associated with Runge–Kutta methods, an hp framework has been developed for optimal control based on collocation at the LGR quadrature points [9–11,20]. In Ref. [21], it is shown that, by a suitable choice of the mesh, extremely fast (exponential) convergence rates can be achieved by using a different polynomial on each mesh interval and by collocating at the LGR quadrature points in each mesh interval. The hp approach has the advantage that convergence is achieved in cases where a p method would fail while an hp method converges faster than an h method because an hp method achieves the same accuracy using a significantly smaller mesh than would be required using an h method. Finally, in the hp approach, the mesh is adaptively adjusted to locate discontinuities at the boundaries of mesh intervals, and the smoothness of the solution is exploited between the mesh intervals by using higher-degree polynomials. Consequently, optimal control problems can be solved much faster in the newly developed hp framework than in an h approach.

Concurrent with the aforementioned hp Gaussian quadrature collocation methods, a great deal of research has been done recently in the area of *computational optimal guidance and control* where the generation of guidance and control commands "relies extensively on onboard computation" [22] and "are model-based or data-based" [22]. Recent research in computational guidance and control includes Refs. [23–28]. Reference [23] presents a new onboard-implementable, real-time convex optimization-based powered-descent guidance algorithm for planetary pinpoint landing. Reference [24] describes terrestrial flight-test demonstrations of large diverts planned using the so-called guidance for fuel-optimal large diverts (G-FOLD) onboard a vertical-takeoff/vertical-landing rocket. Reference [25] provides a verification of fully numerical predictor-corrector entry guidance (FNPEG) for orbital and suborbital entry and skip entry missions. Reference [26] describes an approach for generating optimal feedback control laws for infinite time problems by solving the corresponding Hamilton–Jacobi–Bellman equation. Reference [27] develops a parallelizable quadratic programming approach for model predictive flight control. Reference [28] develops a real-time trajectory optimization framework for autonomous suspended load operations in outdoor environments. It is seen that from the aforementioned research the integration of computational optimization has become an important aspect of guidance and control.

When solving an optimal control problem computationally using the recently developed hp methodology, a process called mesh refinement [29–33] is implemented where the nonlinear programming problem (NLP) associated with the hp approximation is solved on a series of meshes such that the solution on a given mesh is higher in accuracy than the solution on the preceding mesh. The mesh refinement method terminates when a specified accuracy tolerance has been attained. Because the hp method employs Gaussian quadrature collocation, the integration accuracy is as high as possible relative to the number of collocation points. A key aspect of mesh refinement is that the majority of time required by the mesh refinement process is spent determining the mesh itself. It is noted, however, that once a mesh has been determined to a desired accuracy, the solution to the hp discrete approximation of the optimal control problem can be computed quickly.

The research described in this paper is not a mesh refinement method. Mesh refinement is an offline procedure that is performed only once on the full horizon before the execution of the motion of the system, whereas this paper develops an hp Gaussian quadrature collocation method for use online in guidance and control in applications. In particular, in this research, an hp mesh truncation and remapping method is developed over a shrinking horizon. The approach of this paper is to divide the time interval at the end of each guidance cycle into an expired and an unexpired horizon, delete the portion of the mesh associated with the expired horizon, and remap the mesh on the unexpired horizon to a scaled mesh time. In the context of guidance and control, the truncated and remapped mesh is then in close proximity to a mesh that meets the desired accuracy tolerance. The optimal control problem is then solved on the unexpired horizon using this truncated and remapped mesh. In most cases, the solution to the control problem associated with the remapped mesh meets the specified accuracy tolerance, whereas in a few cases a small mesh adjustment is used to meet the tolerance. After solving the optimal control problem on a given horizon, the computed optimal control is then implemented in the system over a time interval, called a guidance cycle, that is significantly shorter than the unexpired horizon. At the end of the guidance cycle, the optimal control problem is solved on the remaining horizon (where now the remaining horizon has shrunk relative to the start of the previous guidance cycle), taking into account new information based on the evolution of the actual system over the previous guidance cycle. In this research, the hp form of the LGR Gaussian quadrature orthogonal collocation method [6–8,29–34] is employed as implemented in the MATLAB optimal control software $\mathbb{GPOPS}-\mathbb{II}$ as described in Ref. [35]. It is noted that $\mathbb{GPOPS}-\mathbb{II}$ is implemented using the sparse NLP solver *IPOPT* [4], where *IPOPT* operates in full-Newton mode. In addition, all first and second derivatives required by the NLP solver are supplied using the MATLAB algorithmic differentiation toolbox *ADiGator* [36,37]. Finally, it is noted that the offline mesh refinement (i.e., before the execution of the motion) and any minimal mesh adjustments required online are performed using the hp mesh refinement method of Ref. [33].

It is important to contrast this research with methods for nonlinear model predictive control (MPC) [38–41]. In our approach the full dynamics are employed, and the optimal open loop control is computed for the entire remaining horizon in each guidance cycle. Also, in earlier work [42–44] collocation techniques employing nonlinear programming are used in guidance maneuvers. Different from this prior research, this paper focuses on the development of a hp mesh truncation and remapping method for use in guidance and control (i.e., for online applications).

The contributions of this research are to develop a mesh truncation and remapping strategy for use online in guidance and control applications. These techniques essentially eliminate the need for mesh refinement. The full nonlinear dynamics and all constraints (nonlinear path constraints and nonlinear boundary conditions) can be included on each guidance cycle due to the speed of the method. Because the horizon shrinks with each successive guidance cycle, the hp mesh decreases in size and the computation time required to solve the hp discrete approximation also decreases as the horizon shrinks.

This paper is organized as follows. Section II describes the independent variables used in this research and the context in which these independent variables are used. Section III provides a description of the continuous-time Bolza optimal control problem under consideration in this research. Section IV provides a discussion of the formulation of the Bolza optimal control problem in multiple-interval form and the approximation of the multiple-interval Bolza optimal control problem using LGR collocation. Section V provides a high-level description of how the control obtained from the solution of the Bolza optimal control problem given in Sec. III is used as an input for guidance and control of the system over the current guidance cycle. Section VI provides a detailed description of the computational guidance and control method developed in this research using hp LGR collocation and sparse nonlinear programming. Section VII provides a detailed description of the exclusion or inclusion of a computation time delay, where the time delay is equal to the computational time required to solve the sparse NLP arising from the hp LGR collocation method. Section VIII provides two examples that demonstrate the utility of the computational guidance and control method in both the absence and the presence of the aforementioned time delay. Section IX provides a discussion of the results obtained in Sec. VIII. Finally, Sec. X provides conclusions on this research.

II. Independent Variables

Two different but related independent variables will be used in this study. The first independent variable, t, is called the *elapsed time* and has a domain $t \in [t_0, t_f]$, where t_0 and t_f are the initial and terminal elapsed times, respectively. The second independent variable, τ, is called the *mesh time* and has a domain $\tau \in [-1, +1]$. The elapsed time and the mesh time are related as

$$t \equiv t(\tau, t_0, t_f) = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} \tag{1}$$

These two independent variables are used in different contexts in this study. First, consider the controlled dynamical system

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{a}(\boldsymbol{y}(t), \boldsymbol{u}(t), t) \tag{2}$$

where $\boldsymbol{y}(t) \in \mathbb{R}^{n_y}$ and $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ are the state and control, respectively, of the system, and $\boldsymbol{a}: \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_y}$. When performing explicit simulation (i.e., time-marching) of a system, the elapsed time is used as the independent variable. Second, consider an optimal control problem of the kind presented in Sec. III. When solving an optimal control problem, implicit simulation (i.e., collocation as described in Sec. IV) is used and the simulation is performed using the mesh time as the independent variable. In the case of implicit simulation, the dynamics of the system as given in Eq. (2) are rewritten as

$$\frac{d\boldsymbol{y}}{d\tau} = \frac{t_f - t_0}{2}\boldsymbol{a}(\boldsymbol{y}(\tau), \boldsymbol{u}(\tau), t(\tau, t_0, t_f)) \tag{3}$$

where $t(\tau, t_0, t_f)$ is given by Eq. (1).

III. Bolza Optimal Control Problem

Without loss of generality, consider the following optimal control problem in Bolza form in terms of the mesh time, τ. Determine the state $\boldsymbol{y}(\tau) \in \mathbb{R}^{n_y}$ and the control $\boldsymbol{u}(\tau) \in \mathbb{R}^{n_u}$ on the domain $\tau \in [-1, +1]$, and the terminal time, t_f, that minimize the cost functional

$$\begin{aligned}\mathcal{J} = &\mathcal{M}(\boldsymbol{y}(-1), t_0, \boldsymbol{y}(+1), t_f)\\ &+ \frac{t_f - t_0}{2}\int_{-1}^{+1} \mathcal{L}(\boldsymbol{y}(\tau), \boldsymbol{u}(\tau), t(\tau, t_0, t_f))\, \mathrm{d}\tau\end{aligned} \tag{4}$$

subject to the dynamic constraints

$$\frac{d\boldsymbol{y}}{d\tau} = \frac{t_f - t_0}{2}\boldsymbol{a}(\boldsymbol{y}(\tau), \boldsymbol{u}(\tau), t(\tau, t_0, t_f)) \tag{5}$$

the inequality path constraints

$$\boldsymbol{c}_{\min} \le \boldsymbol{c}(\boldsymbol{y}(\tau), \boldsymbol{u}(\tau), t(\tau, t_0, t_f)) \le \boldsymbol{c}_{\max} \tag{6}$$

and the boundary conditions

$$\begin{bmatrix} \boldsymbol{y}(-1) \\ t_0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_0 \\ \text{Fixed} \end{bmatrix}, \quad \boldsymbol{b}_f(\boldsymbol{y}(+1), t_f) \le \boldsymbol{0} \tag{7}$$

Suppose now that the independent variable $\tau \in [-1, +1]$ is partitioned into a *mesh* consisting of K *mesh intervals* $\mathcal{I}_k = [T_{k-1}, T_k]$, $k = 1, \ldots, K$, where $-1 = T_0 < T_1 < \ldots < T_K = +1$. The mesh intervals have the property that $\bigcup_{k=1}^{K} \mathcal{I}_k = [-1, +1]$. Let $\boldsymbol{y}^{(k)}(\tau)$ and $\boldsymbol{u}^{(k)}(\tau)$ be the state and control in the mesh interval $\mathcal{I}_k$, $k = 1, \ldots, K$. The Bolza optimal control problem of Eqs. (4–7) can then be rewritten as follows. Minimize the cost functional

$$\begin{aligned}\mathcal{J} = &\mathcal{M}(\boldsymbol{y}^{(1)}(-1), t_0, \boldsymbol{y}^{(K)}(+1), t_f)\\ &+ \frac{t_f - t_0}{2}\sum_{k=1}^{K}\int_{T_{k-1}}^{T_k} \mathcal{L}(\boldsymbol{y}^{(k)}(\tau), \boldsymbol{u}^{(k)}(\tau), t(\tau, t_0, t_f))\, \mathrm{d}\tau\end{aligned} \tag{8}$$

subject to the dynamic constraints

$$\begin{aligned}&\frac{d\boldsymbol{y}^{(k)}(\tau)}{d\tau} = \frac{t_f - t_0}{2}\boldsymbol{a}(\boldsymbol{y}^{(k)}(\tau), \boldsymbol{u}^{(k)}(\tau), t(\tau, t_0, t_f)),\\ &(k = 1, \ldots, K)\end{aligned} \tag{9}$$

the path constraints

$$\boldsymbol{c}_{\min} \le \boldsymbol{c}(\boldsymbol{y}^{(k)}(\tau), \boldsymbol{u}^{(k)}(\tau), t(\tau, t_0, t_f)) \le \boldsymbol{c}_{\max}, \qquad (k = 1, \ldots, K) \tag{10}$$

and the boundary conditions

$$\begin{bmatrix} \boldsymbol{y}^{(1)}(-1) - \boldsymbol{y}_0 \\ t_0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \text{Fixed} \end{bmatrix}, \quad \boldsymbol{b}_f(\boldsymbol{y}^{(K)}(+1), t_f) \le \boldsymbol{0} \tag{11}$$

Because the state must be continuous at each interior mesh point, it is required that the condition $\boldsymbol{y}(T_k^-) = \boldsymbol{y}(T_k^+)$, $(k = 1, \ldots, K-1)$ be satisfied at the interior mesh points $(T_1, \ldots, T_{K-1})$.

IV. LGR Collocation Method for Optimal Control

The multiple-interval form of the continuous-time Bolza optimal control problem in Sec. III is discretized using the hp form of the LGR collocation method as described in Refs. [6–8,29–33]. In the hp LGR collocation method, the state of the continuous-time Bolza optimal control problem and its derivative with respect to τ are approximated in $\mathcal{I}_k$, $k \in [1, \ldots, K]$, as

$$\left.\begin{aligned} \boldsymbol{y}^{(k)}(\tau) \approx \boldsymbol{Y}^{(k)}(\tau) = \textstyle\sum_{j=1}^{N_k+1} \boldsymbol{Y}_j^{(k)} \ell_j^{(k)}(\tau) \\ \frac{d\boldsymbol{y}^{(k)}(\tau)}{d\tau} \approx \frac{d\boldsymbol{Y}^{(k)}(\tau)}{d\tau} = \textstyle\sum_{j=1}^{N_k+1} \boldsymbol{Y}_j^{(k)} \frac{d\ell_j^{(k)}(\tau)}{d\tau} \end{aligned}\right\},$$

$$\ell_j^{(k)}(\tau) = \prod_{\substack{l=1 \\ l \neq j}}^{N_k+1} \frac{\tau - \tau_l^{(k)}}{\tau_j^{(k)} - \tau_l^{(k)}} \tag{12}$$

where $\tau \in [-1, +1]$, $\ell_j^{(k)}(\tau)$, $j = 1, \ldots, N_k + 1$, is a basis of Lagrange polynomials, $(\tau_1^{(k)}, \ldots, \tau_{N_k}^{(k)})$ are the LGR [45] collocation points in $\mathcal{I}_k = [T_{k-1}, T_k)$, and $\tau_{N_k+1}^{(k)} = T_k$ is a noncollocated point. Approximating the dynamics and path constraints given in Eqs. (9) and (10) at the N_k LGR points in each mesh interval $\mathcal{I}_k$, $(k = 1, \ldots, K)$, approximating the boundary conditions given in Eq. (11) using the approximations of the initial and terminal time and state, and approximating the integral in Eq. (8) using an N_k-point LGR quadrature in mesh interval $\mathcal{I}_k$, $(k = 1, \ldots, K)$ leads to the following nonlinear programming problem (NLP). Minimize the cost function:

$$\mathcal{J} \approx \mathcal{M}(\boldsymbol{Y}_1^{(1)}, t_0, \boldsymbol{Y}_{N_K+1}^{(K)}, t_f) + \frac{t_f - t_0}{2} \sum_{k=1}^{K} \sum_{j=1}^{N_k} w_j^{(k)} \mathcal{L}(\boldsymbol{Y}_j^{(k)}, \boldsymbol{U}_j^{(k)}, t(\tau_j^{(k)}, t_0, t_f)) \tag{13}$$

subject to the constraints

$$\left.\begin{aligned} \textstyle\sum_{j=1}^{N_k+1} D_{ij}^{(k)} \boldsymbol{Y}_j^{(k)} - \frac{t_f - t_0}{2} \boldsymbol{a}(\boldsymbol{Y}_i^{(k)}, \boldsymbol{U}_i^{(k)}, t(\tau_i^{(k)}, t_0, t_f)) = \boldsymbol{0} \\ \boldsymbol{c}_{\min} \leq \boldsymbol{c}(\boldsymbol{Y}_i^{(k)}, \boldsymbol{U}_i^{(k)}, t(\tau_i^{(k)}, t_0, t_f)) \leq \boldsymbol{c}_{\max} \\ \boldsymbol{Y}_1^{(1)} - \boldsymbol{y}_0 = \boldsymbol{0} \\ t_0 = \text{Fixed} \\ \boldsymbol{b}_f(\boldsymbol{Y}_{N_K+1}^{(K)}, t_f) \leq \boldsymbol{0} \end{aligned}\right\} \quad (i = 1, \ldots, N_k) \tag{14}$$

where $w_j^{(k)}$ are the N_k LGR weights in mesh interval $\mathcal{I}_k$, $k \in [1, \ldots, K]$ and $D_{ij}^{(k)} = d\ell_j^{(k)}(\tau_i^{(k)})/d\tau$, $(i = 1, \ldots, N_k, j = 1, \ldots, N_k + 1)$ are the elements of the $N_k \times (N_k + 1)$ *LGR differentiation matrix* [6] in mesh interval $\mathcal{I}_k$, $k \in [1, \ldots, K]$. It is noted that the continuity in the state at the interior mesh points $(T_1, \ldots, T_{K-1})$ is enforced via the condition $\boldsymbol{Y}_{N_k+1}^{(k)} = \boldsymbol{Y}_1^{(k+1)}$, $(k = 1, \ldots, K - 1)$, but this continuity constraint is eliminated from the NLP by using the *same* variable for both $\boldsymbol{Y}_{N_k+1}^{(k)}$ and $\boldsymbol{Y}_1^{(k+1)}$. Furthermore, we note that $N = \sum_{k=1}^{K} N_k$ is the total number of LGR points on $\tau \in [-1, +1]$. The sparse NLP arising from the hp LGR collocation method as given in Eqs. (13) and (14) has a specific structure that has been studied extensively. In particular, the key elements of the structure of the NLP arising from the hp LGR collocation method can be found in the NLP constraint Jacobian and the Lagrangian Hessian. The structure of these two matrices can be found in Ref. [46] along with extensive details on the sparse structure of the NLP arising from the hp LGR collocation method.

V. Optimal Guidance

Suppose now that $\mathcal{B}_s = (\boldsymbol{y}(t), \boldsymbol{u}(t), t_f)$ is a solution of the Bolza optimal control problem given on the time horizon $t \in [t_0^{(s)}, t_f^{(s)}]$ as given in Eqs. (8–11), where $s \in [1, 2, 3, \ldots, S]$ is the start of guidance cycle $s \in [1, \ldots, S]$, S is the number of guidance cycles (i.e., S is the number of times that the control is updated during the motion), and

$$t_0^{(s)} = t_0 + sG \tag{15}$$

is the initial time of guidance cycle $s \in [1, 2, \ldots, S]$, where G is the guidance cycle duration. Suppose now that the optimal control $\boldsymbol{u}(t)$

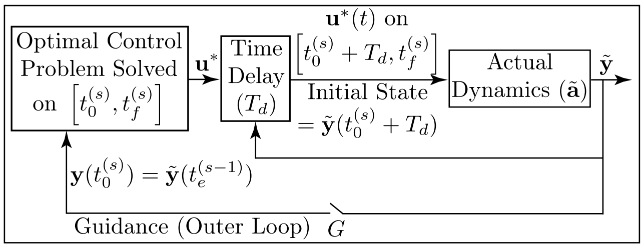

Optimal Control Problem Solved on $[t_0^{(s)}, t_f^{(s)}]$
$\mathbf{u}^*$
Time Delay (T_d)
$\mathbf{u}^*(t)$ on $[t_0^{(s)} + T_d, t_f^{(s)}]$
Initial State $= \tilde{\mathbf{y}}(t_0^{(s)} + T_d)$
Actual Dynamics $(\tilde{\mathbf{a}})$
$\tilde{\mathbf{y}}$
$\mathbf{y}(t_0^{(s)}) = \tilde{\mathbf{y}}(t_e^{(s-1)})$
Guidance (Outer Loop) G

Fig. 1 Optimal guidance and control block diagram.

corresponding to $\mathcal{B}_s$ is implemented in the actual system on a *guidance cycle* $[t_0^{(s)}, t_e^{(s)}]$, where

$$t_e^{(s)} = t_0 + (s + 1)G \tag{16}$$

is the terminal elapsed time of guidance cycle $s \in [1, 2, \ldots, S]$. Because the actual dynamics given in Eq. (17) are different from those of the reference dynamics of Eq. (2) [due, for example, to modeling errors and disturbances], the actual system will evolve according to a different set of dynamics given generically in terms of elapsed time as

$$\frac{d\tilde{\boldsymbol{y}}}{dt} = \tilde{\boldsymbol{a}}(\tilde{\boldsymbol{y}}(t), \boldsymbol{u}^*(t), t) \tag{17}$$

where $\tilde{\boldsymbol{a}}: \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_y}$ is the vector field that represents the actual dynamics and $\tilde{\boldsymbol{y}}(t)$ is the state that arises from the evolution of the actual dynamics. Also, it is important to note that over the first time interval $t \in [t_0, t_0 + G]$ (i.e., when $s = 0$), the motion of the actual system given in Eq. (17) is obtained using the optimal control from the solution of the reference optimal control problem on the time horizon $t \in [t_0, t_f]$. Now, the motion of the actual system differs from that of the reference solution, $\tilde{\boldsymbol{y}}(t) \neq \boldsymbol{y}(t)$, which implies that $\tilde{\boldsymbol{y}}(t_e^{(s)}) \neq \boldsymbol{y}(t_e^{(s)})$ at the end of guidance cycle $s \in [0, 1, 2, \ldots, S]$. This deviation from the optimal solution will not only increase the cost on the actual solution $\tilde{\boldsymbol{y}}(t)$ but also lead to violations in both the path constraints of Eq. (6) and the boundary conditions of Eq. (7). To correct for the deviation from the optimal solution, the Bolza optimal control problem is re-solved on the *unexpired horizon* $[t_e^{(s)}, t_f^{(s)}]$. This shorter-horizon optimal control problem is referred to as the *shrinking-horizon optimal control problem*, and the start time of this shrinking-horizon optimal control problem, denoted a *guidance update time*, is $t_0^{(s)} = t_e^{(s-1)}$. The shrinking-horizon optimal control problem is identical in its mathematical form to the Bolza optimal control problem given in Eqs. (8–11) but requires the following substitutions:

$$\boldsymbol{y}_0 = \boldsymbol{y}(t_0^{(s)}) = \tilde{\boldsymbol{y}}(t_e^{(s-1)}) = \text{Fixed}, \qquad t_0 = t_0^{(s)} = t_e^{(s-1)} = \text{Fixed}, \qquad t_f = t_f^{(s)} = \text{Free} \tag{18}$$

where $t_e^{(s-1)}$ is the elapsed time at the terminus of guidance cycle $s - 1$ and $\tilde{\boldsymbol{y}}(t_e^{(s-1)}) = \tilde{\boldsymbol{y}}(T_e^{(s-1)})$ is the state of the actual system at the terminus of guidance cycle $s - 1$ (where $t_e^{(s-1)}$ and $T_e^{(s-1)}$ represent, respectively, the elapsed and mesh times at the terminus of guidance cycle $s - 1$). For completeness, note again that $t_e^{(s-1)}$ and $T_e^{(s-1)}$ are related via Eq. (1). The process of re-solving the shrinking-horizon optimal control problem on the unexpired horizon $[t_0^{(s)}, t_f^{(s)}]$ together with the motion of the system evolving over a guidance cycle $[t_0^{(s)}, t_e^{(s)}]$ is then repeated for each guidance cycle $s \in [1, 2, \ldots, S]$. Figure 1 provides a block diagram schematic that shows the process of optimal guidance and control for the actual system [i.e., the system defined by the dynamics given in Eq. (17)] over the time interval $t \in [t_0^{(s)}, t_e^{(s)}]$ associated with guidance cycle s. It is noted that Fig. 1 includes a possible time delay T_d, where the time delay is equivalent to the computation time required to solve the optimal control problem on the unexpired horizon (see Sec. VII for further details regarding the inclusion of a time delay in this research).

VI. Computational Method for Optimal Guidance and Control

In this section, a computational method is developed for optimal guidance and control. The approach is to solve the optimal control problem of Sec. V at the start of each guidance cycle using hp LGR collocation together with sparse nonlinear programming. The optimal control obtained from the solution of the sparse nonlinear programming problem (NLP) is then used for both guidance and control over the current guidance cycle. A key aspect of the method described in this paper is to truncate and remap a mesh on which the shrinking-horizon optimal control problem is re-solved at the start of each guidance cycle. The goal of the mesh truncation and remapping strategy is to develop a mesh for which mesh refinement is unnecessary while maintaining a specified solution accuracy tolerance.

To develop the method of this paper, consider again the shrinking-horizon optimal control problem as defined in Eqs. (4–7) but with the substitutions given in Eq. (18). Also, assume that the shrinking-horizon optimal control problem will be solved using hp LGR collocation as described in Sec. IV. Using Fig. 2 as a guide, the mesh truncation and remapping strategy is as follows. Suppose that hp LGR collocation has been used to generate a solution $\mathcal{B}_{s-1}$ of the reduced-horizon optimal control problem for guidance cycle $s-1 \in [0, 1, 2, \ldots, S-1]$ to a desired accuracy as described in Sec. V. The solution $\mathcal{B}_{s-1}$ carries with it a mesh composed of the mesh points $T_k^{(s-1)} \in [-1, +1]$, mesh intervals $\mathcal{I}_k = [T_{k-1}^{(s-1)}, T_k^{(s-1)}]$, and collocation points $(\tau_1^{(s-1)}, \ldots, \tau_{N^{(s-1)}}^{(s-1)})$, where

$$N^{(s-1)} = \sum_{k=1}^{K^{(s-1)}} N_k^{(s-1)} \tag{19}$$

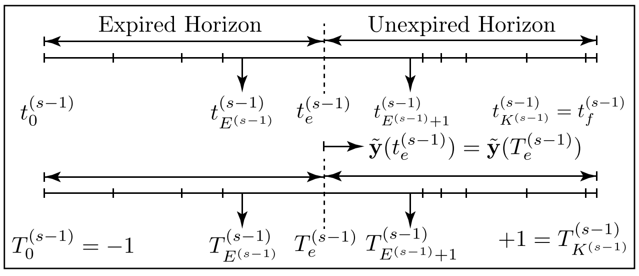
and $N_k^{(s-1)}$ is the number of collocation points in mesh interval $k \in [1, \ldots, N_k^{(s-1)}]$ as defined in Sec. IV. The mesh points $T_k^{(s-1)} \in [-1, +1]$ have corresponding elapsed time points $t_k^{(s-1)} \in [t_0^{(s-1)}, t_f^{(s-1)}]$, $k \in [1, \ldots, K^{(s-1)}]$, where the mesh points $T_k^{(s-1)}$ and the elapsed time points $t_k^{(s-1)}$ are related though Eq. (1), that is,

$$t_k^{(s-1)} = \frac{t_f^{(s-1)} - t_0^{(s-1)}}{2} T_k^{(s-1)} + \frac{t_f^{(s-1)} + t_0^{(s-1)}}{2}, \quad (k = 1, \ldots, N^{(s-1)}) \tag{20}$$

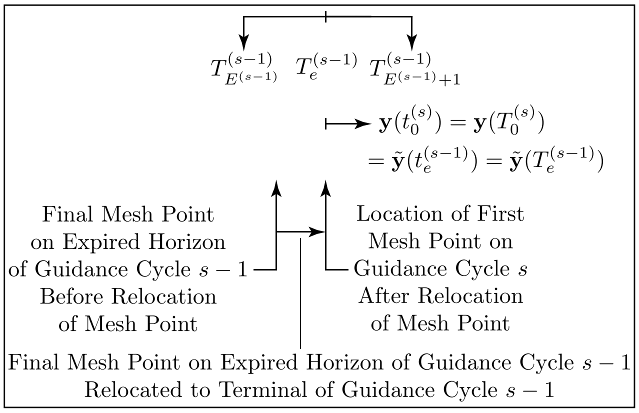
Given the mesh that was used to generate a solution to the optimal control problem using hp LGR collocation on guidance cycle $s-1$, the following approach is used to generate a mesh for use on guidance cycle s. Figure 2 provides a schematic of the approach used to truncate and remap the mesh from the previous guidance cycle $s-1$ to the current guidance cycle s. First, let the domain from guidance cycle $s-1$ be divided into the following two parts. The portions of the mesh domain $\tau \in [-1, +1]$ corresponding to $[-1, T_e^{(s-1)}]$ and $[T_e^{(s-1)}, +1]$ are the *expired horizon* and *unexpired horizon*, respectively, as defined in Sec. V. It can be seen from Fig. 2 that there exist $E^{(s-1)}$ mesh intervals, $\mathcal{I}_k^{(s-1)} = [T_{k-1}^{(s-1)}, T_k^{(s-1)}]$, $(k = 1, \ldots, E^{(s-1)})$, that lie within the expired horizon of guidance cycle $s-1$, while the remaining $K^{(s-1)} - E^{(s-1)}$ mesh intervals, $\mathcal{I}_k^{(s-1)} = [T_{k-1}^{(s-1)}, T_k^{(s-1)}]$, $(k = E^{(s-1)}+1, \ldots, K^{(s-1)})$, lie within the unexpired horizon of guidance cycle $s-1$. It is important to note that the mesh time $T_e^{(s-1)}$ is *not* a mesh point on the mesh domain $\tau \in [-1, +1]$ but in general lies between the two mesh points $T_{E^{(s-1)}}^{(s-1)}$ and $T_{E^{(s-1)}+1}^{(s-1)}$. Thus, the choice of the duration of a guidance cycle is independent of the mesh intervals.

Given the division of the time interval into the expired and unexpired horizons on guidance cycle $s-1$, the goal is to generate a mesh for use on the current guidance cycle $s \in [1, 2, 3, \ldots, S]$ such that the mesh used on guidance cycle s consists of mesh points that are aligned with the mesh points from the unexpired horizon of the previous guidance cycle $s-1$. Because the first $E^{(s-1)}$ mesh intervals, $\mathcal{I}_k^{(s-1)} = [T_{k-1}^{(s-1)}, T_k^{(s-1)}]$, $(k = 1, \ldots, E^{(s-1)})$, lie within the expired horizon, it is reasonable to consider the mesh on guidance cycle s to be constructed using the portion of the mesh on guidance cycle $s-1$ that consists of the unexpired horizon plus the last mesh point on the expired horizon [i.e., the portion of the mesh consisting of points $(T_{E^{(s-1)}}^{(s-1)}, \ldots, T_{K^{(s-1)}}^{(s-1)})$]. Now, it is seen from Fig. 2a that the mesh points $T_{E^{(s-1)}+1}^{(s-1)}, \ldots, T_{K^{(s-1)}}^{(s-1)}$ that lie within the unexpired horizon are already aligned with the expected locations for use on guidance cycle s. Note, however, that because $T_e^{(s-1)}$ is not a mesh point but, instead, lies between the mesh points $T_{E^{(s-1)}}^{(s-1)}$ and $T_{E^{(s-1)}+1}^{(s-1)}$, the mesh point $T_{E^{(s-1)}}^{(s-1)}$ is not aligned with those mesh points corresponding to the unexpired horizon (in general, $T_{E^{(s-1)}}^{(s-1)} < T_e^{(s-1)} < T_{E^{(s-1)}+1}^{(s-1)}$). To bring the first mesh point into alignment with the unexpired horizon for use on guidance cycle s, the final mesh point on the expired horizon, $T_{E^{(s-1)}}^{(s-1)}$, must be relocated to be equal to $T_e^{(s-1)}$. The relocation of the mesh point $T_{E^{(s-1)}}^{(s-1)}$ to the time $T_e^{(s-1)}$ is shown in Fig. 2b. Once the mesh point $T_{E^{(s-1)}}^{(s-1)}$ has been relocated to $T_e^{(s-1)}$, the mesh for use on guidance cycle s can be transformed to the domain $\tau \in [-1, +1]$ for use with the LGR collocation as described in Sec. IV. The transformation of the mesh points $(T_{E^{(s-1)}}^{(s-1)}, \ldots, T_{K^{(s-1)}}^{(s-1)})$ to the time interval $\tau \in [-1, +1]$ is accomplished via the transformation

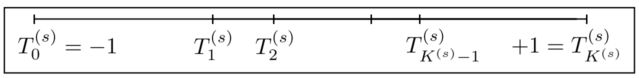
$$T_k^{(s)} = \frac{2\left(T_{k+E^{(s-1)}-1}^{(s-1)} - T_{E^{(s-1)}}^{(s-1)}\right) - \left(1 - T_{E^{(s-1)}}^{(s-1)}\right)}{1 - T_{E^{(s-1)}}^{(s-1)}}, \quad (k = 1, \ldots, K^{(s-1)} - E^{(s-1)} + 1) \tag{21}$$


Expired Horizon
Unexpired Horizon
$t_0^{(s-1)}$
$t_{E^{(s-1)}}^{(s-1)}$
$t_e^{(s-1)}$
$t_{E^{(s-1)}+1}^{(s-1)}$
$t_{K^{(s-1)}}^{(s-1)} = t_f^{(s-1)}$
$\tilde{\mathbf{y}}(t_e^{(s-1)}) = \tilde{\mathbf{y}}(T_e^{(s-1)})$
$T_0^{(s-1)} = -1$
$T_{E^{(s-1)}}^{(s-1)}$
$T_e^{(s-1)}$
$T_{E^{(s-1)}+1}^{(s-1)}$
$+1 = T_{K^{(s-1)}}^{(s-1)}$

a) Expired and unexpired horizons on guidance cycle s–1


$T_{E^{(s-1)}}^{(s-1)}$
$T_e^{(s-1)}$
$T_{E^{(s-1)}+1}^{(s-1)}$
$\mathbf{y}(t_0^{(s)}) = \mathbf{y}(T_0^{(s)})$
$= \tilde{\mathbf{y}}(t_e^{(s-1)}) = \tilde{\mathbf{y}}(T_e^{(s-1)})$
Final Mesh Point on Expired Horizon of Guidance Cycle $s-1$ Before Relocation of Mesh Point
Location of First Mesh Point on Guidance Cycle s After Relocation of Mesh Point
Final Mesh Point on Expired Horizon of Guidance Cycle $s-1$ Relocated to Terminal of Guidance Cycle $s-1$

b) Final mesh point on the expired horizon of guidance cycle s–1 as shown in Fig. 2a relocated to the terminus of guidance cycle s–1



c) Mesh on guidance cycle s, where $T_0^{(s)} = -1$ and $T_{K^{(s)}}^{(s)} = +1$ correspond, respectively, to $T_e^{(s-1)}$ and $T_{K^{(s-1)}}^{(s-1)}$ from guidance cycle s–1

Fig. 2 Schematic for mesh truncation and remapping strategy as part of the computational method for guidance and control using hp LGR collocation.

It is seen from Eq. (21) that the mesh on guidance cycle s is defined on the domain $\tau \in [-1, +1]$ because

$$T_1^{(s)} = \frac{2\left(T_{E^{(s-1)}}^{(s-1)} - T_{E^{(s-1)}}^{(s-1)}\right) - \left(1 - T_{E^{(s-1)}}^{(s-1)}\right)}{1 - T_{E^{(s-1)}}^{(s-1)}} = -\frac{1 - T_{E^{(s-1)}}^{(s-1)}}{1 - T_{E^{(s-1)}}^{(s-1)}} = -1 \tag{22}$$

$$\begin{aligned} T_{K^{(s-1)}-E^{(s-1)}+1}^{(s)} &= \frac{2\left(T_{K^{(s-1)}}^{(s-1)} - T_{E^{(s-1)}}^{(s-1)}\right) - \left(1 - T_{E^{(s-1)}}^{(s-1)}\right)}{1 - T_{E^{(s-1)}}^{(s-1)}} \\ &= \frac{2\left(1 - T_{E^{(s-1)}}^{(s-1)}\right) - \left(1 - T_{E^{(s-1)}}^{(s-1)}\right)}{1 - T_{E^{(s-1)}}^{(s-1)}} = +1 \end{aligned} \tag{23}$$

The number of LGR collocation points used in each mesh interval, $\mathcal{I}^{(s)} = [T_{k-1}^{(s)}, T_k^{(s)}],\ \ k \in [1, \ldots, K^{(s)})]$ on guidance cycle s corresponds to the number of LGR points that were used in the corresponding mesh interval on guidance cycle $s-1$, that is,

$$N_k^{(s)} = N_{k+E^{(s-1)}-1}^{(s-1)}, \qquad (k = 1, \ldots, K^{(s-1)} - E^{(s-1)} + 1) \tag{24}$$

Now, it is noted that, with the exception of the first mesh point, every mesh point of the mesh developed for use on guidance cycle s is aligned with the mesh associated with the unexpired horizon on guidance cycle $s-1$. Moreover, with the exception of the first mesh interval, the LGR collocation points associated with every mesh interval of the shrinking-horizon on guidance cycle s are exactly aligned with the LGR collocation points associated with the unexpired horizon on guidance cycle $s-1$. With regard to the first mesh interval, the relocated mesh point (i.e., $T_{E^{(s-1)}}^{(s-1)}$ on guidance cycle $s-1$ or, equivalently, $T_1^{(s)} = -1$ on guidance cycle s) results in a compression of the LGR points on the first mesh interval of the mesh for use on guidance cycle s. Although the first mesh interval of the mesh used on guidance cycle s has been compressed due to the need to relocate the first mesh point from guidance cycle $s-1$, the mesh is still nearly exactly aligned with the mesh associated with the unexpired horizon on guidance cycle $s-1$. The mesh on guidance cycle s then consists of the $K^{(s)} = K^{(s-1)} - E^{(s-1)} + 1$ mesh intervals, $\mathcal{I}^{(s)} = [T_{k-1}^{(s)}, T_k^{(s)}],\ k \in [1, \ldots, K^{(s)}]$, together with the $N_k^{(s)}$ LGR collocation points in each mesh interval as defined in Eq. (24). It is seen that the mesh on guidance cycle s is smaller and reduced in comparison to the mesh used on guidance cycle $s-1$.

Using the aforementioned procedure to produce the mesh on guidance cycle s from the unexpired horizon on guidance cycle $s-1$, the computational guidance and control method of this paper consists of the following steps. First, the solution on the unexpired horizon of guidance cycle $s-1$ is truncated and remapped for use on guidance cycle s as described above and shown in Fig. 2. This interpolated solution is then used as the initial guess for solving the sparse NLP associated with hp LGR collocation on guidance cycle s. In addition, the boundary conditions given in Eq. (7) [again, with the substitutions given in Eq. (18)] are updated and included as constraints in the sparse NLP. As the time horizon is shrinking, the mesh on guidance cycle s is smaller than the mesh was on guidance cycle $s-1$, and the sparse NLP that is solved on guidance cycle s is also smaller than the sparse NLP that was solved on guidance cycle $s-1$. After the shrinking-horizon optimal control problem is solved on guidance cycle s, the dynamics are simulated over the current guidance cycle $t \in [t_0^{(s)}, t_e^{(s)}]$. In the absence of a time delay (i.e., $T_d = 0$), the control on the horizon $t \in [t_0^{(s)}, t_f^{(s)}]$ is then used as an input to simulate the actual dynamics on $t \in [t_0^{(s)}, t_e^{(s)}]$ (i.e., guidance cycle s). In the presence of a time delay (i.e., $T_d \neq 0$), the control on the horizon $t \in [t_0^{(s-1)}, t_f^{(s-1)}]$ (which is obtained on guidance cycle $s-1$) is used to simulate the actual dynamics on $t \in [t_0^{(s)}, t_0^{(s)} + T_d]$, while the control on the horizon $t \in [t_0^{(s)}, t_f^{(s)}]$ (which is obtained on guidance cycle s) is used to simulate actual dynamics on $t \in [t_0^{(s)} + T_d, t_e^{(s)}]$. After the dynamics evolve over the current guidance cycle s, the mesh is again truncated and remapped for use on the next guidance cycle. This cycle continues, as seen in Fig. 1, until terminal conditions are met. For completeness, Sec. VII provides more information on the exclusion or inclusion of the computation time delay. The method for computational optimal guidance and control is then repeated for every guidance cycle $s \in [1, \ldots, S]$.

VII. Inclusion of Computation Time Delay

The optimal control generated as described in Sec. VI can be employed either in the absence or the presence of a time delay. In the absence of a time delay, it is assumed that the sparse NLP associated with hp LGR collocation can be solved instantaneously, and thus the optimal control for use on the current guidance cycle is available for use at the start of the current guidance cycle. In the presence of a time delay, it is assumed more realistically that the sparse NLP cannot be solved instantaneously, and thus the optimal control for use on the current guidance cycle is available for use only at the time when the sparse NLP has returned a solution. Denoting C as the computation time required to solve the sparse NLP arising from hp LGR collocation on the horizon $t \in [t_0^{(s)}, t_f^{(s)}]$, let $T_d = C$ be the time delay. Because the solution on guidance cycle s will not be available until the time $t = t_0^{(s)} + T_d$, the control obtained on the previous horizon $t \in [t_0^{(s-1)}, t_f^{(s-1)}]$ is used from $t \in [t_0^{(s)}, t_0^{(s)} + T_d]$ (i.e., the control obtained on guidance cycle $s-1$ is used until the solution on guidance cycle s has been computed). Upon solving the sparse NLP (which, as indicated, takes T_d time units), the control computed on guidance cycle s is available for use from $t = t_0^{(s)} + T_d$ onward and is used as an input to simulate the actual dynamics on $t \in [t_0^{(s)} + T_d, t_e^{(s)}]$. In the case where a time delay is absent, it is assumed that $T_d = 0$, and the control computed on guidance cycle s is available from $t = t_0^{(s)}$ onward.

VIII. Examples

Two examples are now considered that employ the method for guidance and control as described in Sec. VI. Both examples are considered in the absence and in the presence of a time delay. The guidance cycle duration for the first example is $G = 30$ s, whereas the guidance cycle duration for the second example, which is a four-phase optimal control problem, is $G = 30$ s in the first three phases and $G = 5$ s in the fourth phase. The guidance cycles were chosen to be small enough that the vehicle can accurately attain the terminal conditions, but large enough that it is possible to solve the optimal control problem in less time than the guidance cycle duration. In the case where a time delay is absent, the actual dynamics of Eq. (17) are simulated in elapsed time t on $[t_0^{(s)}, t_e^{(s)}]$ (the current guidance cycle) using a cubic spline interpolation of the time series of the hp LGR collocation control that has been computed on the current unexpired horizon $[t_0^{(s)}, t_f^{(s)}]$ with the initial conditions obtained from the simulation of the actual dynamics at the terminus of the previous guidance cycle $s-1$. In the case where a time delay is present, the actual dynamics of Eq. (17) are simulated in elapsed time t on $[t_0^{(s)}, t_0^{(s)} + T_d]$ using a cubic spline interpolation of the time series of the hp LGR collocation control that was computed on the *previous* unexpired horizon $[t_0^{(s-1)}, t_f^{(s-1)}]$, and simulated in elapsed time t on $[t_0^{(s)} + T_d, t_e^{(s)}]$ using a cubic spline interpolation of the time series of the hp LGR collocation control that was computed on the *current* unexpired horizon $[t_0^{(s)}, t_f^{(s)}]$. Regardless of whether a time delay is absent or present, the initial state $\boldsymbol{y}(t_0^{(s)}) = \boldsymbol{y}(T_0^{(s)}) = \boldsymbol{y}_0^{(s)}$ [see Fig. 2c] used at the start of the current guidance cycle s is equal to the state $\tilde{\boldsymbol{y}}(t_e^{(s-1)}) = \tilde{\boldsymbol{y}}(T_e^{(s-1)})$ [see Fig. 2a] obtained from the simulation of the actual dynamics at the terminus of the previous guidance cycle $s-1$ [see Fig. 1, where it is shown that $\boldsymbol{y}(t_0^{(s)}) = \tilde{\boldsymbol{y}}(t_e^{(s-1)})$]. All results shown in the section were obtained using the MATLAB optimal control software $\mathbb{GPOPS} - \mathbb{II}$ [35] with the nonlinear programming problem (NLP) solver *IPOPT* [4], where *IPOPT* was employed in full-Newton (second-derivative) mode using a warm start on each guidance cycle. For both examples, the reference optimal solutions and all guidance solutions were computed using an NLP solver tolerance, ϵ_{NLP}, of 10^{-7}. In addition, all reference optimal solutions were obtained using a mesh accuracy tolerance, ϵ_{mesh}, of 10^{-7}. With regard to the guidance and control solutions, for the first example, the mesh accuracy tolerance used for the guidance updates was set to 10^{-7}, whereas for the second example, the mesh accuracy tolerance used for the guidance updates was set to 10^{-5}. Finally, the perturbed dynamics were simulated using the MATLAB differential equation solver ode113 using a relative error tolerance of 10^{-5}. Furthermore, in addition to the mesh truncation and remapping strategy described in Sec. VI, any necessary mesh refinement was performed using the mesh refinement method described in Ref. [33] using a minimum of three and maximum of 14 allowable collocation points per interval. To employ *IPOPT* in full-Newton mode, all required first and second derivatives were obtained using the open-source algorithmic differentiation software *ADiGator* [37]. Finally, all computations shown in this section were on a 3.1 GHz Intel Core i7 MacBook Pro running Mac OS-X version 10.12.6 (Sierra) with 16 GB 1867 MHz DDR3 RAM and MATLAB Version R2016a (build 9.0.0.341360), and all computation (CPU) times, denoted C, are in reference to this aforementioned machine.

A. Example 1: Reusable Launch Vehicle Entry

Consider the problem of steering a reusable launch vehicle from an entry condition to a terminal state while maximizing the cross-range during entry [1]. The problem is to minimize the objective functional

$$J = -\phi(t_f) \tag{25}$$

subject to the dynamic constraints

$$\begin{aligned}
&\dot{r} = v\sin\gamma, \qquad \dot{\theta} = \frac{v\cos\gamma\sin\psi}{r\cos\phi},\\
&\dot{\phi} = \frac{v\cos\gamma\cos\psi}{r}, \qquad \dot{v} = -D - g\sin\gamma,\\
&\dot{\gamma} = \frac{L\cos\sigma}{v} - \left(\frac{g}{v} - \frac{v}{r}\right)\cos\gamma, \qquad \dot{\psi} = \frac{L\sin\sigma}{v\cos\gamma} + \frac{v\cos\gamma\sin\psi\tan\phi}{r}
\end{aligned} \tag{26}$$

the boundary conditions

$$\begin{aligned}
t_0 = t_0^{(s)}, \qquad (r(t_0), r(t_f)) = (r_0^{(s)}, r_f),\\
\theta(t_0) = \theta_0^{(s)}, \qquad \phi(t_0) = \phi_0^{(s)}, \qquad (v(t_0), v(t_f)) = (v_0^{(s)}, v_f),\\
(\gamma(t_0), \gamma(t_f)) = (\gamma_0^{(s)}, \gamma_f), \qquad \psi(t_0) = \psi_0^{(s)}
\end{aligned} \tag{27}$$

where r is the geocentric radius, θ is the longitude, ϕ is the latitude, v is the speed, γ is the flight path angle, ψ is the azimuth angle, α is the angle of attack, σ is the bank angle, $L = qSC_L/m$ is the lift specific force, $D = qSC_D/m$ is the drag specific force, $q = \rho v^2/2$ is the dynamic pressure, $\rho = \rho_0 \exp(-h/H)$ is the atmospheric density, ρ_0 is the sea level density, $h = r - R_e$ is the altitude, R_e is the radius of the Earth, H is the density scale height, $g = \mu/r^2$ is the acceleration due to gravity, and μ is the gravitational parameter. In this example, the initial conditions $(r_0^{(s)}, \theta_0^{(s)}, \phi_0^{(s)}, v_0^{(s)}, \gamma_0^{(s)}, \psi_0^{(s)})$ change depending upon the initial time $t_0^{(s)}$ where $s \in [0, 1, 2, \ldots, S]$, and S is the number of guidance cycles. Finally, the initial conditions corresponding to the full-horizon (reference) optimal solution are given as

$$\begin{aligned}
(t_0^{(0)}, t_f) &= (0, \text{Free})\ \text{s},\\
(r(t_0^{(0)}), r(t_f)) &= (79248, 24384)\ \text{m} + R_e,\\
(\theta(t_0^{(0)}), \theta(t_f)) &= (0, \text{Free})\ \text{deg},\\
(\phi(t_0^{(0)}), \phi(t_f)) &= (0, \text{Free})\ \text{deg},\\
(v(t_0^{(0)}), v(t_f)) &= (7802.88, 762)\ \text{m/s},\\
(\gamma(t_0^{(0)}), \gamma(t_f)) &= (-1, -5)\ \text{deg},\\
(\psi(t_0^{(0)}), \psi(t_f)) &= (90, \text{Free})\ \text{deg}
\end{aligned} \tag{28}$$

and t_f corresponds to the final time obtained on the reference optimal solution on the entire horizon $t \in [t_0^{(0)}, t_f^{(0)}]$.

The computational method for optimal guidance and control developed in this paper is analyzed using the following model that is perturbed from the reference dynamics of Eq. (26). Specifically, in the perturbed model the lift, L, and the drag, D, in Eq. (26) are replaced with

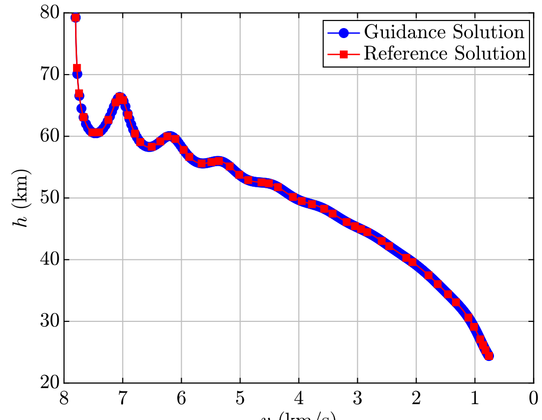
$$\tilde{L} = q_1 L, \qquad \tilde{D} = q_2 D \tag{29}$$

where (q_1, q_2) are random variables drawn on each guidance cycle from a uniform distribution on the interval $[a, b]$.

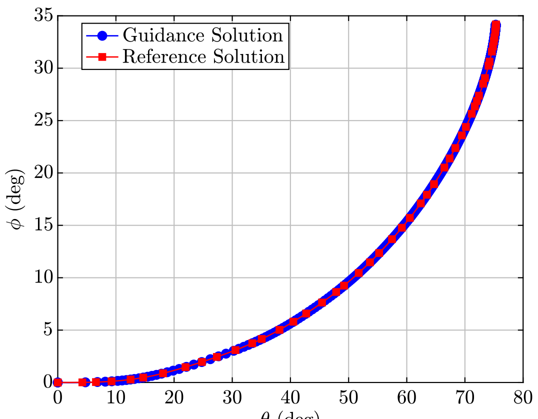
1. *Results Using Computational Optimal Guidance and Control Method in the Absence of a Time Delay*

Suppose now that the optimal guidance and control method of this paper operates in the absence of a computation time delay (i.e., $T_d = 0$). Then, starting with the reference initial conditions of Eq. (28), the optimal control problem given in Eqs. (25–27) is re-solved at the start of each guidance cycle $s \in [1, 2, 3, \ldots, S]$ on a shrinking horizon $[t_0^{(s)}, t_f^{(s)}]$ (where $t_0^{(s)} = t_0 + sG$). Furthermore, the simulation of the actual dynamics is performed with the random variables q_1 and q_2 in Eq. (29), each drawn from a uniform distribution $[a, b] = [0.8, 1.2]$. The state obtained by simulating the actual dynamics [i.e., using the perturbed lift and drag given in Eq. (29)] on the intervals $t \in [t_0^{(s)}, t_0^{(s)} + G]$ (where $s \in [0, 1, 2, \ldots, S]$) using the control computed on $[t_0^{(s)}, t_f^{(s)}]$ via hp LGR collocation is shown in Figs. 3a–3d alongside the reference state. In addition, Figs. 3e and 3f show the control used in the simulation of the actual dynamics alongside the reference control. It is seen that the simulated solution on each of the guidance cycles is close to the reference solution even though the model used in the simulation [Eq. (40)] is different from the reference model [Eq. (31)].

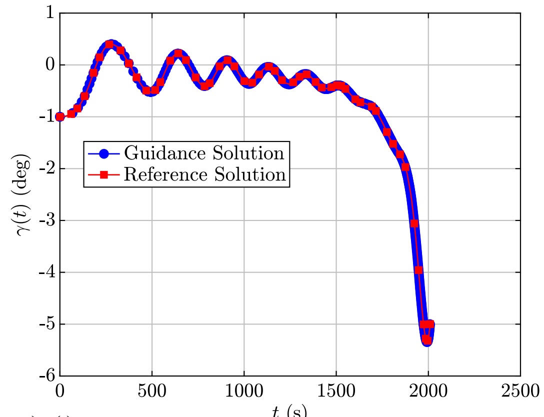
Now let $\Delta\boldsymbol{y}(t) = \tilde{\boldsymbol{y}}(t) - \boldsymbol{y}^*(t)$ be the difference between the state, $\tilde{\boldsymbol{y}}(t)$, obtained using the actual dynamics [i.e., using the perturbed lift and drag given in Eq. (29)] and the state, $\boldsymbol{y}^*(t)$, obtained from the reference solution of the optimal control problem using hp LGR collocation. Furthermore, let t_f^* be the terminal time obtained from the numerical solution of the optimal control problem using hp LGR collocation over the last guidance cycle. Then the differences $\Delta h(t_f^*)$, $\Delta v(t_f^*)$, and $\Delta\gamma(t_f^*)$ are given, respectively, as $(\Delta h(t_f^*), \Delta v(t_f^*), \Delta\gamma(t_f^*)) = (-8.6047 \times 10^{-1}\ \text{m}, -7.0626 \times 10^{-2}\ \text{m/s}, 3.92 \times 10^{-6}\ \text{rad})$. It is seen that these differences are, respectively, $\mathcal{O}(10^{-1})$ m, $\mathcal{O}(10^{-2})$ m/s, and $\mathcal{O}(10^{-6})$ rad. Next, Fig. 4 shows the characteristics of the mesh as a function of the guidance cycle, s, where Fig. 4a shows the number of mesh intervals, K, and the number of collocation points, N, whereas Fig. 4b shows the computation time, C, and the number of mesh refinements, M, as a function of the guidance cycle $s \in [1, \ldots, S]$. It is seen that a minimal amount of mesh refinement is required through all of the guidance cycles in that nearly all of the solutions are obtained on the first mesh without requiring any mesh refinement. Furthermore, it is seen that the size of the mesh progressively decreases as the guidance cycles evolve such that the last guidance cycle mesh consists of 20 collocation points (i.e., $N = 20$) and only two mesh intervals (i.e., $K = 2$). Thus, for the vast majority of guidance cycles the optimal control problem is not only solved efficiently, but the mesh size decreases, making it possible to re-solve the optimal control problem using a progressively



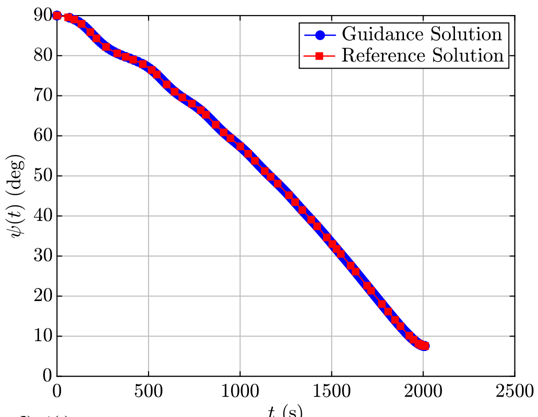
a) ***h* vs. *v***



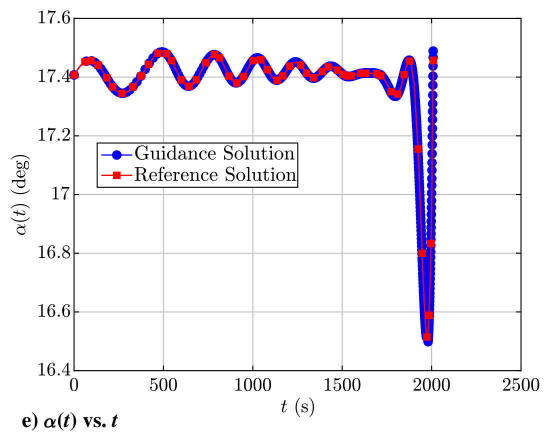
b) ***ϕ* vs. *θ***



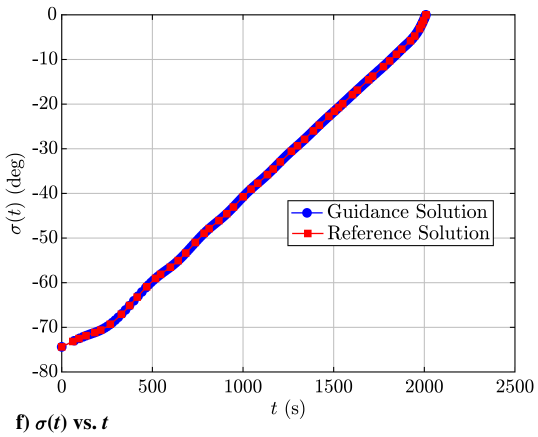
c) ***γ*(*t*) vs. *t***



d) ***ψ*(*t*) vs. *t***

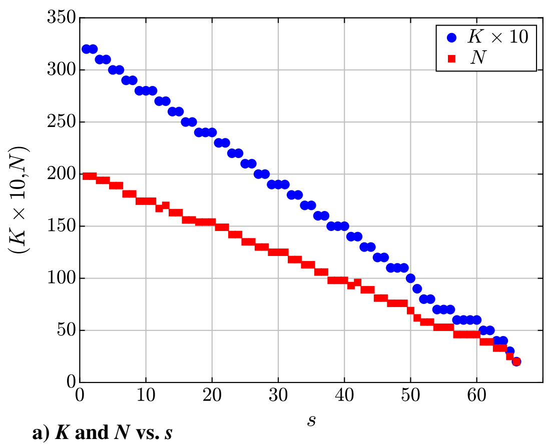


e) ***α*(*t*) vs. *t***

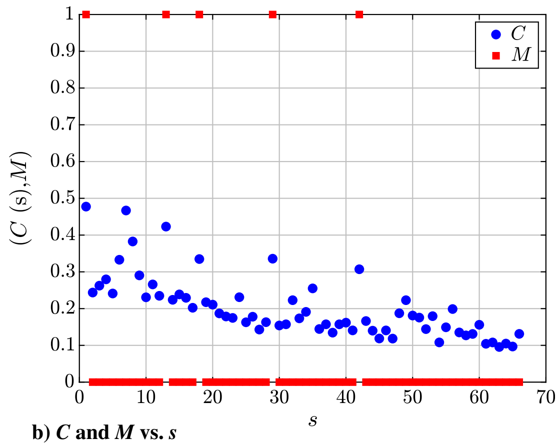


f) ***σ*(*t*) vs. *t***

Fig. 3 Solution obtained via simulation of the actual dynamics [i.e., using the perturbed lift and drag given in Eq. (29)] for Example 1 on each guidance cycle in the absence of a time delay.

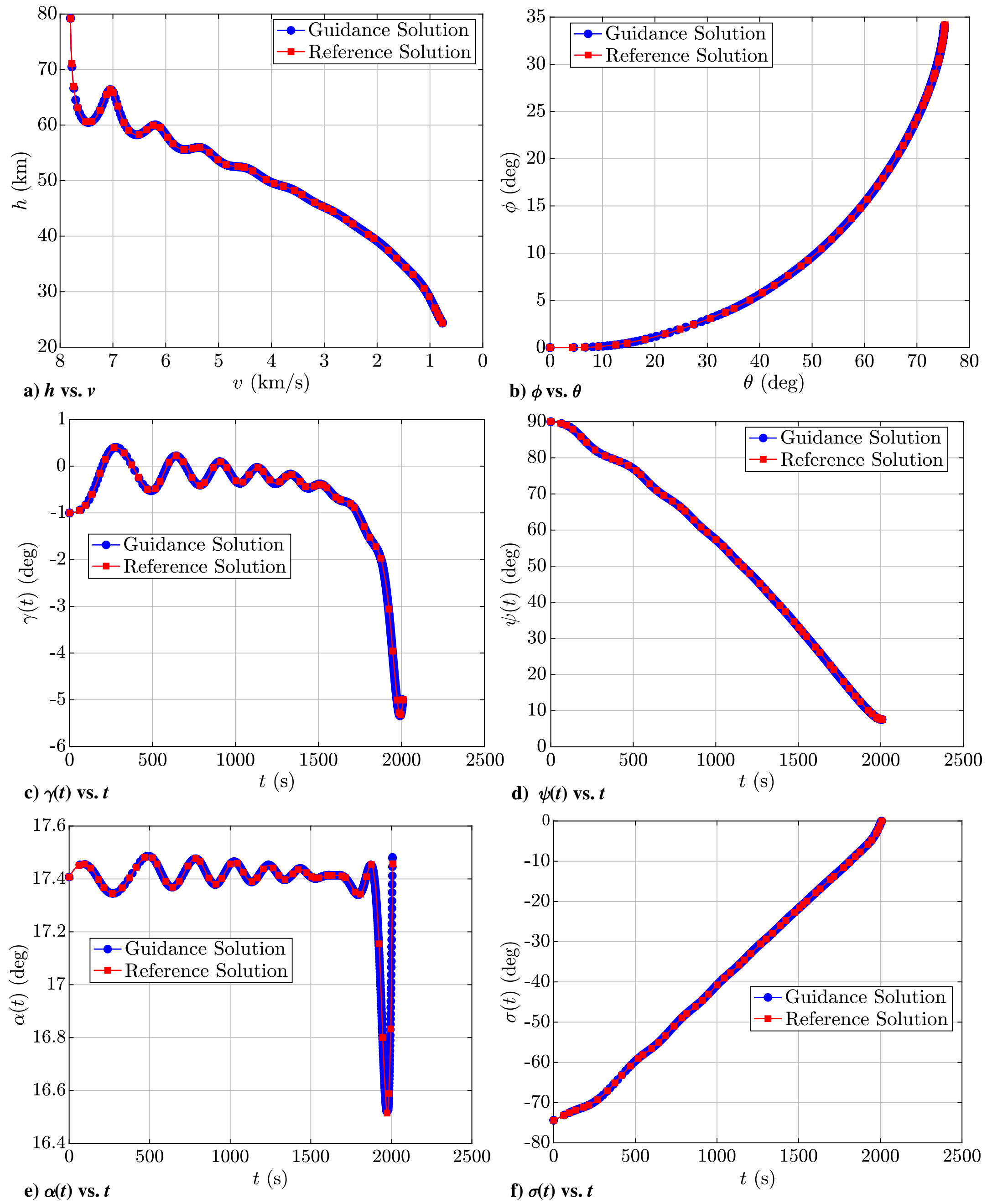


a) ***K* and *N* vs. *s***



b) ***C* and *M* vs. *s***

Fig. 4 Mesh characteristics for Example 1 on each guidance cycle using *hp* LGR collocation method in the absence of a time delay.



a) *h* vs. *v* **b) ϕ vs. θ** **c) $\gamma(t)$ vs. *t*** **d) $\psi(t)$ vs. *t*** **e) $\alpha(t)$ vs. *t*** **f) $\sigma(t)$ vs. *t***

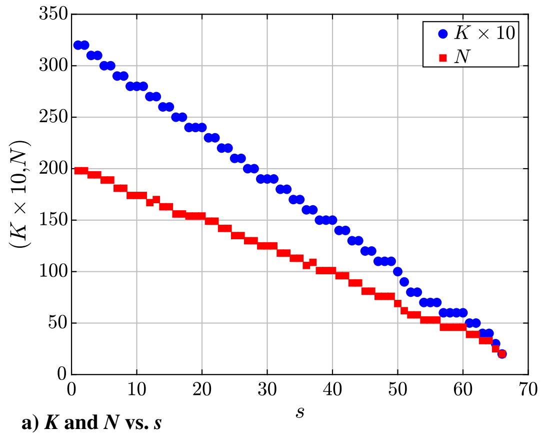
Fig. 5 Solution obtained via simulation of the actual dynamics given [i.e., using the perturbed lift and drag given in Eq. (29)] for Example 1 on each guidance cycle in the presence of a time delay.

smaller finite-dimensional approximation of the optimal control problem.

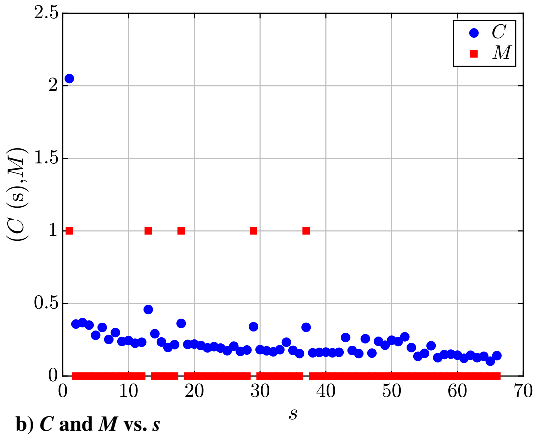
2. *Results Using Computational Optimal Guidance and Control Method in the Presence of a Time Delay*

Suppose now that the optimal guidance and control method of this paper operates in the presence of a computation time delay (i.e., $T_d \neq 0$). Then, starting with the reference initial conditions of Eq. (28), the optimal control problem given in Eqs. (25–27) is re-solved at the start of each guidance cycle $s \in [1, 2, 3, \ldots, S]$ on a shrinking horizon $[t_0^{(s)}, t_f^{(s)}]$ (where $t_0^{(s)} = t_0 + sG$). Furthermore, the simulation of the actual dynamics is performed with the random variables q_1 and q_2 in Eq. (29) each drawn from a uniform distribution $[a, b] = [0.8, 1.2]$. The state obtained by simulating the actual dynamics [i.e., using the perturbed lift and drag given in Eq. (29)] on the intervals $t \in [t_0^{(s)}, t_0^{(s)} + G]$ (where $s \in [0, 1, 2, \ldots, S]$) via hp LGR collocation is shown in Figs. 5a–5d alongside the reference state. In addition, Figs. 5e and 5f show the control used in the simulation of the actual dynamics alongside the reference control. It is seen that the simulated solution on each of the guidance cycles is close to the reference solution even though the model used in the simulation [Eq. (40)] is different from the reference model [Eq. (31)].

Next, and similar to the approach used to analyze the results in the absence of a time delay, let $\Delta \mathbf{y}(t) = \tilde{\mathbf{y}}(t) - \mathbf{y}^*(t)$ be the difference between the state $\tilde{\mathbf{y}}(t)$ obtained using the actual dynamics [i.e., using the perturbed lift and drag given in Eq. (29)] and the state $\mathbf{y}^*(t)$ obtained from the reference solution of the optimal control problem using hp LGR collocation. Furthermore, let t_f^* be the terminal time obtained from the numerical solution of the optimal control problem using hp LGR collocation over the last guidance cycle. Then the differences $\Delta h(t_f^*)$, $\Delta v(t_f^*)$, and $\Delta\gamma(t_f^*)$ are given, respectively, as $(\Delta h(t_f^*), \Delta v(t_f^*), \Delta\gamma(t_f^*)) = (-2.0341 \times 10^{-1}\ \text{m}, -1.9212 \times 10^{-2}\ \text{m/s}, 2.35 \times 10^{-5}\ \text{rad})$. It is seen that these differences are,



a) K and N vs. s



b) C and M vs. s

Fig. 6 Mesh characteristics for Example 1 on each guidance cycle using hp LGR collocation method in the presence of a time delay.

respectively, $\mathcal{O}(10^{-1})$ m, $\mathcal{O}(10^{-2})$ m/s, and $\mathcal{O}(10^{-5})$ rad. Next, Fig. 6 shows the characteristics of the mesh as a function of the guidance cycle, s. Specifically, Fig. 6a shows the number of mesh intervals, K, and the number of collocation points, N, while Fig. 6b shows the computation time, C, and the number of mesh refinements, M, as a function of the guidance cycle $s \in [1, \ldots, S]$. It is seen that a minimal amount of mesh refinement is required through all of the guidance cycles in that nearly all of the solutions are obtained on the first mesh without requiring any mesh refinement. Furthermore, it is seen that the size of the mesh progressively decreases as the guidance cycles evolve such that the last guidance cycle mesh consists of 20 collocation points (i.e., $N = 20$) and only 2 mesh intervals (i.e., $K = 2$). Thus, for the vast majority of guidance cycles, not only is the optimal control problem solved efficiently, but also the mesh size decreases, making it possible to re-solve the optimal control problem using a progressively smaller finite-dimensional approximation of the optimal control problem.

B. Example 2: Reusable Launch Vehicle Ascent

Consider the following four-phase optimal control problem of steering a reusable launch vehicle from an initial condition to an entry condition while maximizing the fuel at the terminal conditions [47,48]. The problem is to minimize the objective functional

$$J = -m^{(4)}(t_f) \tag{30}$$

subject to the dynamic constraints in each phase,

$$\begin{aligned}
\dot{r} &= v \sin\gamma, \quad \dot{\theta} = \frac{v\cos\gamma\cos\psi}{r\cos\phi}, \quad \dot{\phi} = \frac{v\cos\gamma\sin\psi}{r},\\
\dot{v} &= \frac{F_T}{m} - g\sin\gamma + \omega^2 r\cos\phi(\sin\gamma\cos\phi - \cos\gamma\sin\phi\sin\psi),\\
v\dot{\gamma} &= \frac{F_N}{m} w_1 - g\cos\gamma + \frac{v^2}{r}\cos\gamma + 2\omega v\cos\phi\cos\psi\\
&\quad + \omega^2 r\cos\phi(\cos\gamma\cos\phi + \sin\gamma\sin\phi\sin\psi),\\
v\dot{\psi} &= \frac{F_N}{m\cos\gamma} w_2 - \frac{v^2}{r}\cos\gamma\cos\psi\ \tan\phi\\
&\quad + 2\omega v(\tan\gamma\cos\phi\sin\psi - \sin\phi) - \frac{\omega^2 r}{\cos\gamma}\sin\phi\cos\phi\cos\psi,\\
\dot{m} &= -\frac{P}{g_0 I_{sp}}
\end{aligned} \tag{31}$$

the path constraint in each phase

$$w_1^2 + w_2^2 = 1 \tag{32}$$

the path constraint in the fourth phase

$$\alpha_{\min}^{(4)} \le \alpha^{(4)} \le \alpha_{\max}^{(4)} \tag{33}$$

the linkage constraints on the time and state between phases 1, 2, and 3,

$$\begin{bmatrix}
t_f^{(p)} - t_0^{(p+1)} & r(t_f^{(p)}) - r(t_0^{(p+1)})\\
\theta(t_f^{(p)}) - \theta(t_0^{(p+1)}) & \phi(t_f^{(p)}) - \phi(t_0^{(p+1)})\\
v(t_f^{(p)}) - v(t_0^{(p+1)}) & \gamma(t_f^{(p)}) - \gamma(t_0^{(p+1)})\\
\psi(t_f^{(p)}) - \psi(t_0^{(p+1)}) & m(t_f^{(p)}) - m(t_0^{(p+1)})
\end{bmatrix} = \begin{bmatrix} 0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0 \end{bmatrix},$$
$$p \in [1, 2, 3] \tag{34}$$

and the boundary conditions

$$\begin{aligned}
&t_0 = t_0^{(s)},\ (r(t_0), \quad r(t_f)) = (r_0^{(s)}, r_f),\\
&\theta(t_0) = \theta_0^{(s)}, \qquad \phi(t_0) = \phi_0^{(s)},\ (v(t_0), v(t_f)) = (v_0^{(s)}, v_f),\\
&(\gamma(t_0), \gamma(t_f)) = (\gamma_0^{(s)}, \gamma_f), \qquad \psi(t_0) = \psi_0^{(s)}, \qquad m(t_0) = m_0^{(s)},\\
&i(t_f) = i_f
\end{aligned} \tag{35}$$

where r is the geocentric radius, θ is the Earth-relative longitude, ϕ is the geocentric latitude, v is the Earth-relative speed, γ is the Earth-relative flight path angle, ψ is the Earth-relative heading angle, m is the mass, i is the orbital inclination, α is the angle of attack and is the first control, and (w_1, w_2) form the second and third controls such that w_1 and w_2 represents, respectively, the cosine and the sine of the bank angle, σ [i.e., the bank angle σ is not used as the control, but w_1 and w_2 are used as controls subject to the first path constraint of Eq. (32)]. It is noted that the bank angle can be computed a posteriori using w_1 and w_2 as

$$\sigma = \tan^{-1}(-w_2, -w_1) + \pi \tag{36}$$

where $\tan^{-1}(\cdot, \cdot)$ is the four-quadrant inverse tangent, and negating w_1 and w_2 in Eq. (36) ensures that $\sigma \in [0, 2\pi]$. Furthermore, $h = r - R_e$ is the altitude, where $R_e = 6,378,166$ m is the radius of the Earth; $F_T = F_T(P, \alpha, D)$ is the sum of the tangential components of the lift, drag, and thrust; and $F_N = F_N(P, \alpha, L)$ is sum of the normal components of the lift, drag, and thrust. It is noted that the lift and drag are obtained from two-dimensional polynomial fits of tabular data as given in Ref. [47]. In this example, the initial conditions $(r_0^{(s)}, \theta_0^{(s)}, \phi_0^{(s)}, v_0^{(s)}, \gamma_0^{(s)}, \psi_0^{(s)}, m_0^{(s)})$ change depending upon the initial time $t_0^{(s)}$ where $s \in [0, 1, 2, \ldots, S]$. Finally, the initial conditions corresponding to the full-horizon (reference) optimal solution are given as

$$
\begin{aligned}
t_0^{(0)} &= 126\ \text{s}, & r(t_0^{(0)}) &= 46447.6\,\text{m} + R_e,\\
\theta(t_0^{(0)}) &= -120.74\ \text{deg}, & \phi(t_0^{(0)}) &= 34.1\ \text{deg},\\
v(t_0^{(0)}) &= 1384.7\ \text{m/s}, & &\\
\gamma(t_0^{(0)}) &= 26.4\ \text{deg}, & \psi(t_0^{(0)}) &= 259\ \text{deg},\\
m(t_0^{(0)}) &= 675702.8\ \text{kg} & &
\end{aligned} \tag{37}
$$

the terminal conditions are given as

$$
\begin{aligned}
&r(t_f) = 203720\ \text{m} + R_e, \qquad \theta(t_f) = \text{Free},\\
&\phi(t_f) = \text{Free}, \quad v(t_f) = 7782\ \text{m/s}, \quad \gamma(t_f) = 0\ \text{deg},\\
&\psi(t_f) = \text{Free}, \quad m(t_f) = \text{Free}, \quad i(t_f) = 98\ \text{deg}
\end{aligned} \tag{38}
$$

and the limits on the angle of attack in the fourth phase are

$$
\alpha_{\min}^{(4)} = 0.1\ \text{deg}, \qquad \alpha_{\max}^{(4)} = 15\ \text{deg} \tag{39}
$$

where t_f corresponds to the final time obtained on the reference optimal control solution on the entire horizon $t \in [t_0^{(0)}, t_f^{(0)}]$.

The computational method for optimal guidance and control developed in this paper is analyzed using the following model that is perturbed from the reference dynamics. Specifically, in the perturbed model the forces F_T and F_N in Eq. (31) are replaced with

$$
\tilde{F}_T = q_1 F_T, \qquad \tilde{F}_N = q_2 F_N \tag{40}
$$

where (q_1, q_2) are random variables drawn on each guidance cycle from a uniform distribution on the interval $[a, b]$, and the bank angle,

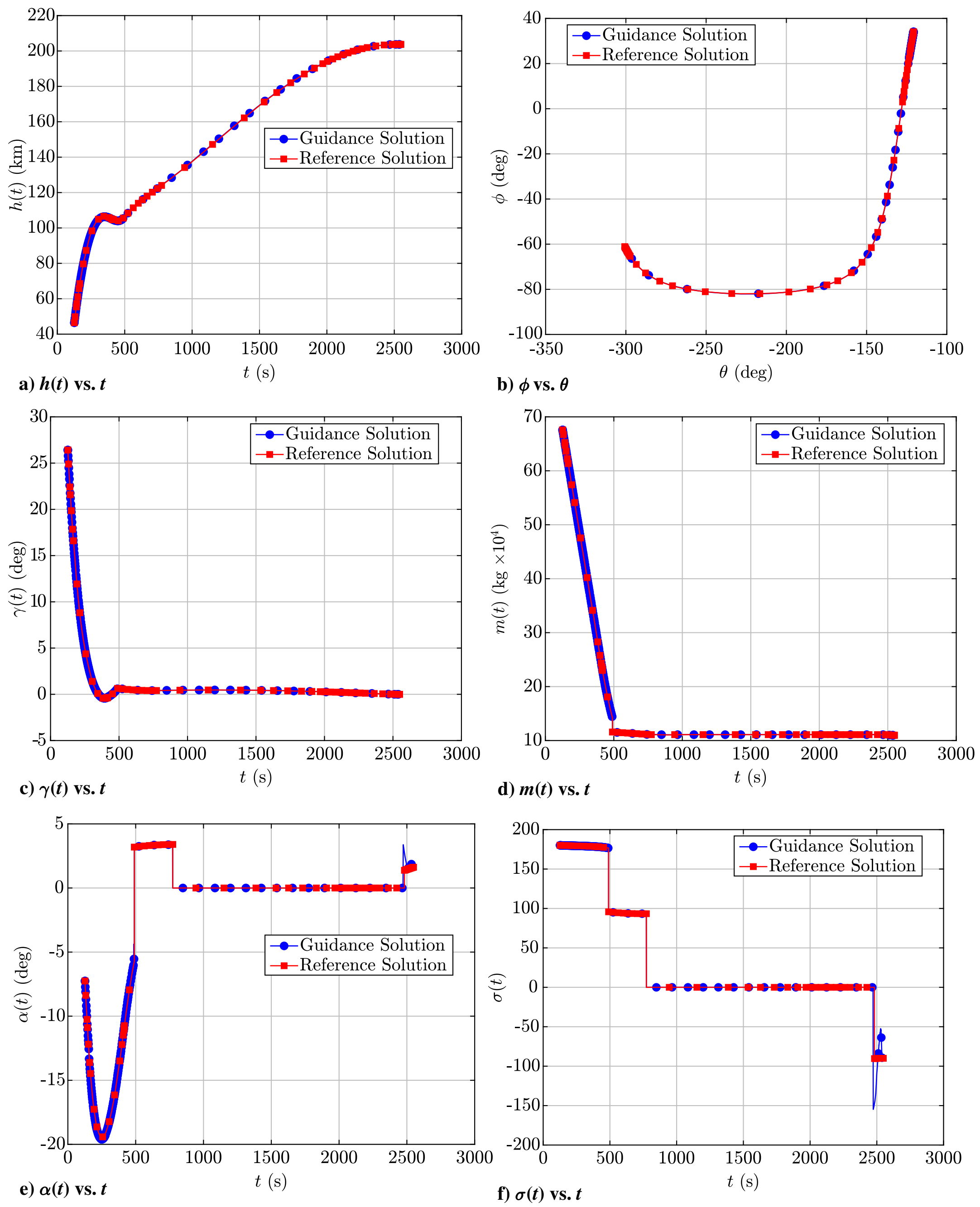


Fig. 7 Solution obtained via simulation of the actual dynamics [i.e., using the perturbed model given in Eq. (40)] for Example 2 on each guidance cycle in the absence of a time delay.

σ, is obtained from Eq. (36) using w_1^* and w_2^*, where w_1^* and w_2^* are obtained by solving the sparse NLP associated with hp LGR collocation over the unexpired horizon.

1. Results Using Computational Optimal Guidance and Control Method in the Absence of a Time Delay

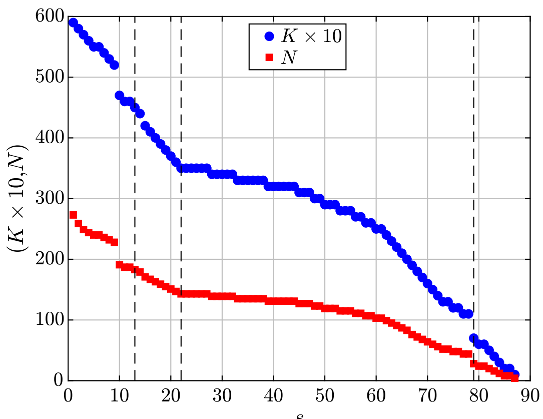
Suppose now that the optimal guidance and control method of this paper operates in the absence of a computation time delay (i.e., $T_d = 0$). Then, starting with the reference initial conditions of Eq. (37), the optimal control problem given in Eqs. (30–35) is re-solved at the start of each guidance cycle $s \in [1, 2, 3, \ldots, S]$ on a shrinking horizon $[t_0^{(s)}, t_f^{(s)}]$ (where $t_0^{(s)} = t_0 + sG$). Furthermore, the simulation of the actual dynamics is performed with the random variables q_1 and q_2 in Eq. (40) each drawn from a uniform distribution $[a, b] = [0.9, 1.1]$. The state obtained by simulating the actual dynamics [i.e., using the perturbed model given in Eq. (40)] on the intervals $t \in [t_0^{(s)}, t_0^{(s)} + G]$ (where $s \in [0, 1, 2, \ldots, S]$) using the control computed on $[t_0^{(s)}, t_f^{(s)}]$ via hp LGR collocation is shown in Figs. 7a–7d alongside the reference state. In addition, Figs. 7e and 7f show the control used in the simulation of the actual dynamics alongside the reference control. It is seen that the simulated solution on each of the guidance cycles is close to the reference solution even though the model used in the simulation is different from the reference model.

Now let $\Delta \boldsymbol{y}(t) = \tilde{\boldsymbol{y}}(t) - \boldsymbol{y}^*(t)$ be the difference between the state, $\tilde{\boldsymbol{y}}(t)$, obtained using the perturbed dynamics [i.e., using the perturbed model given in Eq. (40)] and the state, $\boldsymbol{y}^*(t)$, obtained from the reference solution of the optimal control problem using hp LGR collocation. Furthermore, let t_f^* be the terminal time obtained from the numerical solution of the optimal control problem using hp LGR collocation over the last guidance cycle. Then the differences $\Delta h(t_f^*)$, $\Delta v(t_f^*)$, $\Delta\gamma(t_f^*)$, and $\Delta i(t_f^*)$ are given, respectively, as $(\Delta h(t_f^*), \Delta v(t_f^*), \Delta\gamma(t_f^*), \Delta i(t_f^*)) = (-3.6452 \times 10^{-4}\ \text{m}, -1.264 \times 10^{-1}\ \text{m/s}, -2.07 \times 10^{-8}\ \text{rad}, -4.4409 \times 10^{-15}\ \text{rad})$. It is seen that these differences are, respectively, $\mathcal{O}(10^{-4})$ m, $\mathcal{O}(10^{-1})$ m/s, $\mathcal{O}(10^{-8})$ rad, and $\mathcal{O}(10^{-15})$ rad. Next, Fig. 8 shows the characteristics of the mesh as a function of the guidance cycle, s. Specifically, Fig. 8a shows the number of mesh intervals, K, and the number of collocation points, N, while Fig. 8b shows the computation time, C, and the number of mesh refinements, M, as a function of the guidance cycle $s \in [1, \ldots, S]$. It is seen that no mesh refinement is required through all of the guidance cycles in that all of the solutions are obtained on the first mesh without requiring any mesh refinement. Furthermore, it is seen that the size of the mesh progressively decreases as the guidance cycles evolve such that in the last guidance cycle the mesh consists of four collocation points (i.e., $N = 4$) and only a single mesh interval (i.e., $K = 1$). Thus, for the vast majority of guidance cycles, not only is the optimal control problem solved efficiently, but also the mesh size decreases, making it possible to re-solve the optimal control problem using a progressively smaller finite-dimensional approximation of the optimal control problem.

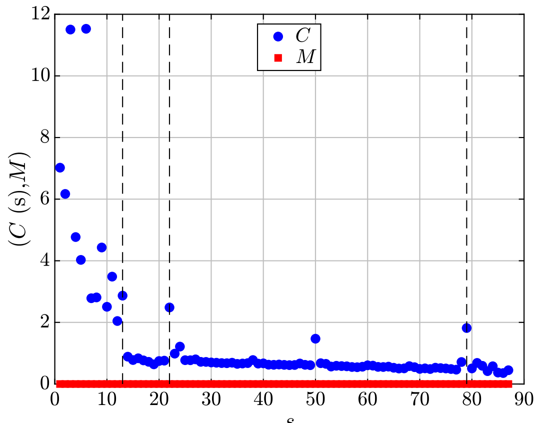
2. Results Using Computational Optimal Guidance and Control Method in the Presence of a Time Delay

Suppose now that the optimal guidance and control method of this paper operates in the presence of a computation time delay (i.e., $T_d \neq 0$). Then, starting with the reference initial conditions of Eq. (37), the optimal control problem given in Eqs. (30–35) is re-solved at the start of each guidance cycle $s \in [1, 2, 3, \ldots, S]$ on a shrinking horizon $[t_0^{(s)}, t_f^{(s)}]$ (where $t_0^{(s)} = t_0 + sG$). Furthermore, the simulation of the actual dynamics is performed with the random variables q_1 and q_2 in Eq. (40) each drawn from a uniform distribution $[a, b] = [0.9, 1.1]$. The state obtained by simulating the actual dynamics [i.e., using the perturbed model given in Eq. (40)] on the intervals $t \in [t_0^{(s)}, t_0^{(s)} + G]$ (where $s \in [0, 1, 2, \ldots, S]$) using the control computed on $[t_0^{(s)}, t_f^{(s)}]$ via hp LGR collocation is shown in Figs. 9a–9d alongside the reference state. In addition, Figs. 9e and 9f show the control used in the simulation of the actual dynamics alongside the reference control. It is seen that the simulated solution on each of the guidance cycles is close to the reference solution even though the model used in the simulation is different from the reference model.

Now let $\Delta \boldsymbol{y}(t) = \tilde{\boldsymbol{y}}(t) - \boldsymbol{y}^*(t)$ be the difference between the state $\tilde{\boldsymbol{y}}(t)$ obtained using the perturbed model [i.e., using the model given in Eq. (40)] and the state $\boldsymbol{y}^*(t)$ obtained from the reference solution of the optimal control problem using hp LGR collocation. Furthermore, let t_f^* be the terminal time obtained from the numerical solution of the optimal control problem using hp LGR collocation over the last guidance cycle. Then the differences $\Delta h(t_f^*)$, $\Delta v(t_f^*)$, $\Delta\gamma(t_f^*)$, and $\Delta i(t_f^*)$ are given, respectively, as $(\Delta h(t_f^*), \Delta v(t_f^*), \Delta\gamma(t_f^*), \Delta i(t_f^*)) = (-4.6885 \times 10^{-3}\ \text{m}, -4.985 \times 10^{-1}\ \text{m/s}, -2.67 \times 10^{-7}\ \text{rad}, 4.4408 \times 10^{-15}\ \text{rad})$. It is seen that these differences are, respectively, $\mathcal{O}(10^{-3})$ m, $\mathcal{O}(10^{-1})$ m/s, $\mathcal{O}(10^{-7})$ rad, and $\mathcal{O}(10^{-15})$ rad. Next, Fig. 10 shows the characteristics of the mesh as a function of the guidance cycle, s. Specifically, Fig. 10a shows the number of mesh intervals, K, and the number of collocation points, N, while Fig. 10b shows the computation time, C, and the number of mesh refinements, M, as a function of the guidance cycle $s \in [1, \ldots, S]$. It is seen that no mesh refinement is required through all of the guidance cycles. Furthermore, it is seen that the size of the mesh progressively decreases in size as the guidance cycles evolve such that in the last two guidance cycles the mesh consists of four collocation points (i.e., $N = 4$) and only a single mesh interval (i.e., $K = 1$). Thus, for the vast majority of guidance cycles, not only is the optimal control problem solved efficiently, but also the mesh size decreases, making it possible to re-solve the optimal control problem using a progressively smaller finite-dimensional approximation of the optimal control problem.

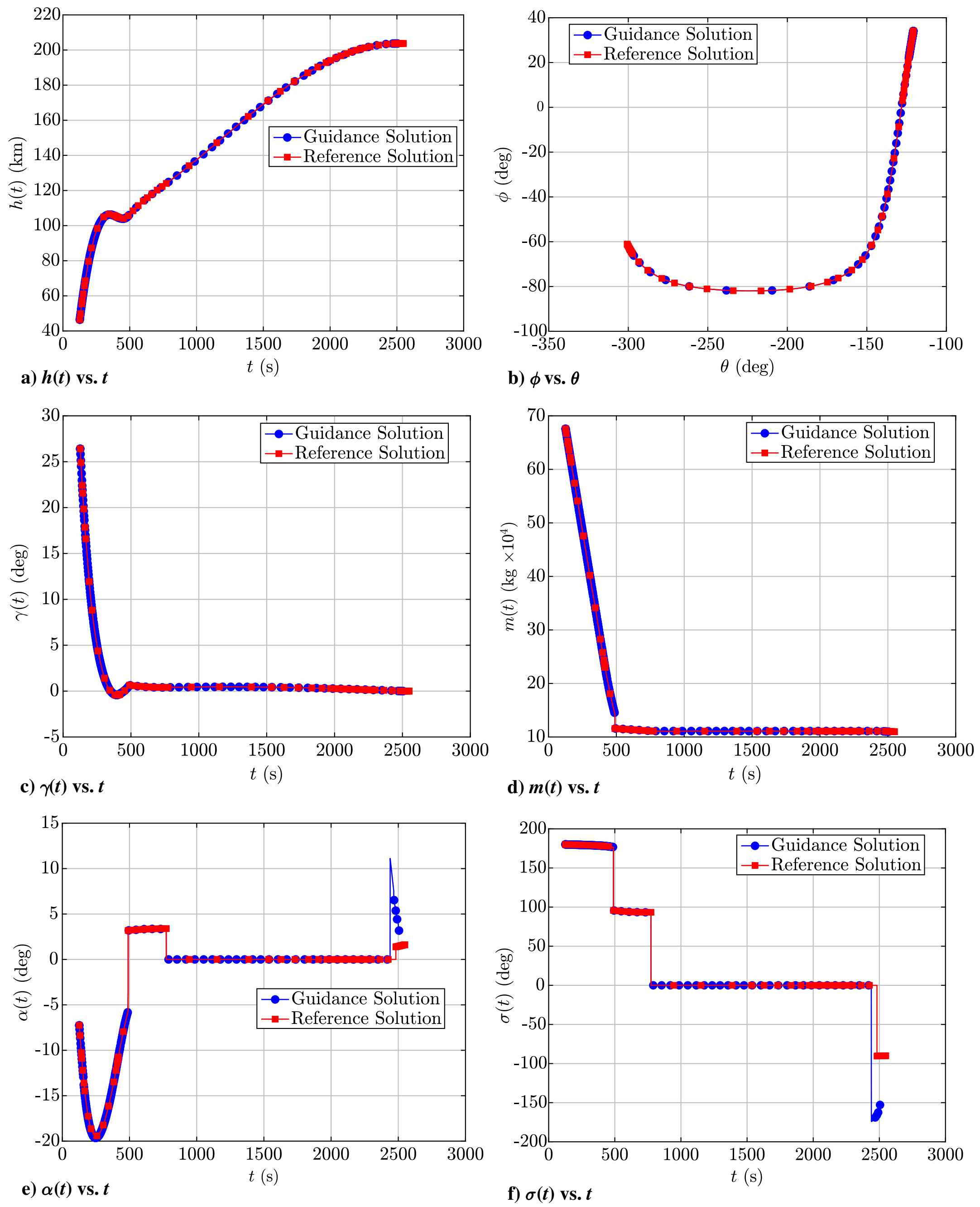


a) K and N vs. s



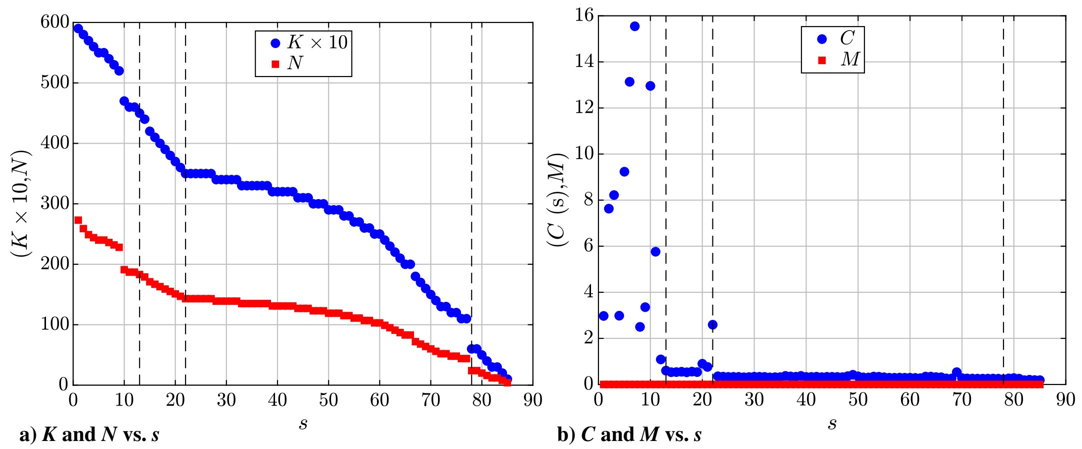
b) C and M vs. s

Fig. 8 Mesh characteristics for Example 2 on each guidance cycle using hp LGR collocation method in the absence of a time delay.



a) ***h*(*t*) vs. *t***　b) ***ϕ* vs. *θ***　c) ***γ*(*t*) vs. *t***　d) ***m*(*t*) vs. *t***　e) ***α*(*t*) vs. *t***　f) ***σ*(*t*) vs. *t***

Fig. 9 Solution obtained via simulation of the actual dynamics [i.e., using the perturbed model given in Eq. (40)] for Example 2 on each guidance cycle in the presence of a time delay.



a) ***K* and *N* vs. *s***　b) ***C* and *M* vs. *s***

Fig. 10 Mesh characteristics for Example 2 on each guidance cycle using *hp* LGR collocation method in the presence of a time delay.

IX. Discussion

The two examples analyzed in Sec. VIII illustrate the key features of the computational optimal guidance and control method developed in this paper. The first key feature pertains to the size and placement of the mesh as the horizon shrinks. As a result, the method developed in this paper is designed such that the truncated and remapped mesh for use on the current unexpired horizon is smaller than the mesh that was used to solve the optimal control problem on the previous unexpired horizon. Also, it is seen in both examples that little to no mesh refinement is required in order to meet the mesh refinement accuracy tolerance. Specifically, it is seen from Figs. 4b and 6b that only 5 of the 66 guidance cycles require any mesh refinement in the absence or presence of a time delay for the reusable launch vehicle entry problem (Example 1). Moreover, it is seen for Example 1 that in the few cases where mesh refinement is required, the mesh refinement accuracy tolerance is satisfied using only a single mesh refinement iteration (where seen from Figs. 4b and 6b that $M = 1$ for 5 of the 66 guidance cycles and is zero otherwise). These results show that the mesh truncation and remapping strategy described in Sec. VI provides a mesh at the start of each guidance cycle that is quite close to a mesh that satisfies the mesh refinement accuracy tolerance. Consequently, in only a few cases, more than one sparse NLP must be solved in order to obtain a control that can be implemented for use on that guidance cycle.

A second important property of the computational optimal guidance method developed in this research pertains to the computational efficiency with which a solution is returned by the NLP solver that satisfies the mesh refinement accuracy tolerance. For the reusable launch vehicle entry problem (Example 1), the guidance cycle was $G = 30$ s. It is seen from Fig. 4b and 6b that in all cases the time required to solve the sparse NLP is significantly shorter than the guidance cycle. Specifically, the maximum time required to re-solve the sparse NLP in Example 1 either in the absence or presence of a time delay is approximately 0.5 s and 2.1 s, respectively (see again Figs. 4b and 6b). Next, for the reusable launch vehicle ascent problem (Example 2), the guidance cycle for the first three phases was $G = 30$ s, and the fourth phase has a guidance cycle of $G = 5$ s. It is seen from Figs. 8b and 10b that, for the guidance cycles in the first phase, the computation time is a significant fraction of the guidance cycle (approximately 12 and 16 s, respectively, in the absence and the presence of a time delay). It is noted, however, that for the second and third phases the computation time is significantly smaller (never greater than 3 s either in the absence or the presence of a time delay). In the fourth phase, where the guidance cycle $G = 5$ s, the computation time is significantly shorter than the guidance cycle. Specifically, the maximum times required to re-solve the sparse NLP in phase four of Example 2 in the absence and presence of a time delay are, respectively, approximately 1.82 s (as shown in Fig. 8b) and 0.25 s (as shown in Fig. 10b). Thus, although a large computation time is required to perform a guidance update during the first phase of the reusable launch vehicle ascent problem, it is still possible to compensate for any modeling errors in the first phase of the motion, because, in the first phase of flight, the vehicle is a sufficiently large distance from the target.

The third key aspect of the computational method for guidance and control developed in this paper is the accuracy with which the terminal constraints are attained via simulation of the actual (perturbed) dynamics. For the reusable launch vehicle entry problem (Example 1), the attained terminal conditions differ from the specified terminal conditions in both the absence and the presence of a time delay by slightly less than 1 m and 1 m/s in altitude and speed, respectively, and by significantly less than 1 deg in flight path angle. Next, for the reusable launch vehicle ascent problem (Example 2), the attained terminal conditions differ from the specified terminal conditions in both the absence and the presence of a time delay by significantly less than 1 m in altitude, slightly less than 1 m/s in speed, and much less than 1 deg in both flight path angle and inclination. These results indicate that the method developed in this research is capable of compensating for the significant perturbations in the dynamic model relative to the reference model.

Another key point, although somewhat subtle, pertains to the manner in which the sparse NLP is solved. In this study, the highly capable NLP solver *IPOPT* was used in full-Newton (second-derivative) mode. Note, however, that *IPOPT* employs an interior-point method. Although interior-point methods work extremely well in full-Newton mode, one limitation of an interior-point method is that the initial guess for the NLP solver, which in this case is accurate and is based on a cubic spline interpolant of the solution on the unexpired horizon, is generally perturbed away from the starting point. Consequently, even using an accurate initial guess, it is generally the case that an interior-point method will need to execute several major iterations before it converges to a solution on the unexpired horizon. As a result, the large computation times seen on the guidance cycles in the first phase of the reusable launch vehicle ascent problem are a result of the interior-point method in *IPOPT*. It is likely that optimization techniques such as a full-Newton sequential quadratic programming (SQP) method could perform better given the initial guesses that are used in a method such as the one described in this research.

X. Conclusions

A method has been developed for computational optimal guidance and control using adaptive Gaussian quadrature collocation and sparse nonlinear programming. The method employs adaptive Legendre–Gauss–Radau (LGR) quadrature collocation together with sparse nonlinear programming in order to numerically solve a successively shrinking-horizon optimal control problem. A key aspect of the method is that on any guidance cycle the mesh truncation and remapping strategy results in a mesh that is smaller than the mesh used on the previous guidance cycle. Moreover, because the mesh on the previous guidance cycle satisfied the mesh refinement accuracy tolerance, these mesh points are well placed for rapidly solving the LGR quadrature collocation approximation of the optimal control problem on the unexpired horizon. Using the reduced size mesh, the LGR quadrature collocation sparse nonlinear programming problem is solved to generate the control that is used as an input to the system on the current guidance cycle. The computational approach to guidance and control developed in this paper has been demonstrated on two well-known and computationally challenging aerospace flight optimal control problems. For both examples, a perturbed dynamic model from that of the reference model was used to simulate the motion of the system. In addition, the method has been demonstrated in both the absence and the presence of a computational time delay equal to the time required to solve the sparse nonlinear programming problem. The results of this study demonstrate that the method developed in this paper is viable as a computational method for optimal guidance and control.

Acknowledgments

The authors acknowledge support for this research from the U.S. Office of Naval Research under grant N00014-15-1-2048, from the U.S. National Science Foundation under grants DMS-1522629, DMS-1819002, and CMMI-1563225, from the Naval Air Warfare Center Weapons Division, China Lake, California under Contract No. N68936-17-C-0011, and from the U.S. Department of Defense Science, Mathematics and Research for Transformation (SMART) Scholarship for Service Program.

References

[1] Betts, J. T., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed., SIAM Press, Philadelphia, PA, 2009, Chaps. 3, 4.

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

[3] Gill, P. E., Murray, W., and Saunders, M. A., "User's Guide for SNOPT Version 7: Software for Large Scale Nonlinear Programming," Feb. 2006.

[4] Biegler, L. T., and Zavala, V. M., "Large-Scale Nonlinear Programming Using IPOPT: An Integrating Framework for Enterprise-Wide Optimization," *Computers and Chemical Engineering*, Vol. 33, No. 3, March 2009, pp. 575–582.
doi:10.1016/j.compchemeng.2008.08.006

[5] 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

[6] 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

[7] Garg, D., Hager, W. W., and Rao, A. V., "Pseudospectral Methods for Solving Infinite-Horizon Optimal Control Problems," *Automatica*, Vol. 47, No. 4, April 2011, pp. 829–837.
doi:10.1016/j.automatica.2011.01.085

[8] 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," *Computational Optimization and Applications*, Vol. 49, No. 2, June 2011, pp. 335–358.
doi:10.1007/s10589-009-9291-0

[9] Hager, W. W., Hou, H., and Rao, A. V., "Lebesgue Constants Arising in a Class of Collocation Methods," *IMA Journal of Numerical Analysis*, Vol. 13, No. 1, Oct. 2017, pp. 1884–1901.
doi:10.1093/imanum/drw060

[10] Hager, W. W., Hou, H., and Rao, A. V., "Convergence Rate for a Gauss Collocation Method Applied to Unconstrained Optimal Control," *Journal of Optimization Theory and Applications*, Vol. 169, No. 3, 2016, pp. 801–824.
doi:10.1007/s10957-016-0929-7

[11] Hager, W. W., Liu, J., Mohapatra, S., Rao, A. V., and Wang, X.-S., "Convergence Rate for a Gauss Collocation Method Applied to Constrained Optimal Control," *SIAM Journal on Control and Optimization*, Vol. 56, No. 2, 2018, pp. 1386–1411.
doi:10.1137/16M1096761

[12] 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

[13] 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

[14] 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

[15] 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

[16] 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

[17] 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

[18] Hager, W. W., "Runge-Kutta Methods in Optimal Control and the Transformed Adjoint System," *Numerische Mathematik*, Vol. 87, No. 2, 2000, pp. 247–282.
doi:10.1007/s002110000178

[19] Bonnans, J. F., and Laurent-Varin, J., "Computation of Order Conditions for Symplectic Partitioned Runge-Kutta Schemes with Application to Optimal Control," *Numerische Mathematik*, Vol. 103, No. 1, March 2006, pp. 1–10.
doi:10.1007/s00211-005-0661-y

[20] Du, W., Chen, W., Yang, L., and Hager, W. W., "Bounds for Integration Matrices that Arise in Gauss and Radau Collocation," *Computational Optimization and Applications*.
doi:10.1007/s10589-019-00099-5

[21] Hager, W. W., Hou, H., Mohapatra, S., Rao, A. V., and Wang, X.-S., "Convergence Rate for an Radau hp Collocation Method Applied to Constrained Optimal Control," *Computational Optimization and Applications*.
doi:10.1007/s10589-019-00100-1

[22] Lu, P., "Introducing Computational Guidance and Control," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, p. 193.
doi:10.2514/1.G002745

[23] Dueri, D., Açikmeşe, B., Scharf, D. P., and Harris, M. W., "Customized Real-Time Interior-Point Methods for Onboard Powered-Descent Guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 197–212.
doi:10.2514/1.G001480

[24] Scharf, D. P., Açikmeşe, B., Dueri, D., Benito, J., and Casoliva, J., "Implementation and Experimental Demonstration of Onboard Powered-Descent Guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 213–229.
doi:10.2514/1.G000399

[25] Lu, P., Brunner, C. W., Stachowiak, S. J., Mendeck, G. F., and Tigges, M. A., "Verification of a Fully Numerical Entry Guidance Algorithm," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 230–247.
doi:10.2514/1.G000327

[26] Adurthi, N., Singla, P., and Majji, M., "Sparse Approximation-Based Collocation Scheme for Nonlinear Optimal Feedback Control Design," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 248–264.
doi:10.2514/1.G001755

[27] Ferranti, L., and Keviczky, T., "Operator-Splitting and Gradient Methods for Real-Time Predictive Flight Control Design," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 265–277.
doi:10.2514/1.G000288

[28] Torre, G. D. L., Theodorou, E., and Johnson, E. N., "Autonomous Suspended Load Operations via Trajectory Optimization and Variational Integrators," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, March–April 2017, pp. 278–291.
doi:10.2514/1.G001769

[29] Darby, C. L., Hager, W. W., and Rao, A. V., "Anhp-Adaptive Pseudospectral Method for Solving Optimal Control Problems," *Optimal Control Applications and Methods*, Vol. 32, No. 4, July–Aug. 2011, pp. 476–502.
doi:10.1002/oca.v32.4

[30] Darby, C. L., Hager, W. W., and Rao, A. V., "Direct Trajectory Optimization Using a Variable Low-Order Adaptive Pseudospectral Method," *Journal of Spacecraft and Rockets*, Vol. 48, No. 3, May–June 2011, pp. 433–445.
doi:10.2514/1.52136

[31] Patterson, M. A., Hager, W. W., and Rao, A. V., "A ph Mesh Refinement Method for Optimal Control," *Optimal Control Applications and Methods*, Vol. 36, No. 4, July–Aug. 2015, pp. 398–421.
doi:10.1002/oca.v36.4

[32] Liu, F., Hager, W. W., and Rao, A. V., "Adaptive Mesh Refinement for Optimal Control Using Nonsmoothness Detection and Mesh Size Reduction," *Journal of the Franklin Institute*, Vol. 352, No. 10, Oct. 2015, pp. 4081–4106.
doi:10.1016/j.jfranklin.2015.05.028

[33] Liu, F., Hager, W. W., and Rao, A. V., "Adaptive Mesh Refinement for Optimal Control Using Decay Rates of Legendre Polynomial Coefficients," *IEEE Transactions on Control System Technology*, Vol. 26, No. 4, 2018, pp. 1475–1483.
doi:10.1109/TCST.2017.2702122

[34] Francolin, C. C., Benson, D. A., Hager, W. W., and Rao, A. V., "Costate Approximation in Optimal Control Using Integral Gaussian Quadrature Collocation Methods," *Optimal Control Applications and Methods*, Vol. 36, No. 4, July–Aug. 2015, pp. 381–397.
doi:10.1002/oca.v36.4

[35] Patterson, M. A., and Rao, A. V., "$\mathbb{GPOPS}-\mathbb{II}$, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *ACM Transactions on Mathematical Software*, Vol. 41, No. 1, Oct. 2014, pp. 1–37.
doi:10.1145/2684421

[36] Weinstein, M. J., and Rao, A. V., "A Source Transformation via Operator Overloading Method for the Automatic Differentiation of Mathematical Functions in MATLAB," *ACM Transactions on Mathematical Software*, Vol. 42, No. 1, June 2016, pp. 11:1–11:44.
doi:10.1145/2699456

[37] Weinstein, M. J., and Rao, A. V., "Algorithm: ADiGator, a Toolbox for the Algorithmic Differentiation of Mathematical Functions in

MATLAB," *ACM Transactions on Mathematical Software*, Vol. 44, No. 2, Oct. 2017, pp. 1–25.
doi:10.1145/3132683

[38] Mayne, D. Q., "Model Predictive Control: Recent Developments and Future Promise," *Automatics*, Vol. 50, No. 12, Dec. 2014, pp. 2967–2986.
doi:10.1016/j.automatica.2014.10.128

[39] Rawlings, J. B., Mayne, D. Q., and Diehl, M. M., *Model Predictive Control: Theory, Computation, and Design*, 2nd ed., Nob Hill Publishing, 2017.

[40] Chen, H., and Alllgower, F., "A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability," *Automatica*, Vol. 34, No. 10, 1998, pp. 1205–1217.
doi:10.1016/S0005-1098(98)00073-9

[41] Yu, M., Griffith, D. W., and Biegler, L. T., *Nonlinear Programming Formulations for Nonlinear and Economic Model Predictive Control*, Springer International Publishing, Cham, 2019, pp. 465–489.

[42] Sekhavat, P., Fleming, A., and Ross, I. M., "Time-Optimal Nonlinear Feedback Control for the NPSAT1 Spacecraft," *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2005, pp. 843–850.

[43] Ross, I. M., Sekhavat, P., Fleming, A., and Gong, Q., "Optimal Feedback Control: Foundations, Examples, and Experimental Results for a New Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 2, March–April 2008, pp. 307–321.
doi:10.2514/1.29532

[44] Bollino, K., Ross, I. M., and Doman, D., "Optimal Nonlinear Feedback Guidance for Reentry Vehicles," *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2006-6074, Aug. 2006.

[45] Abramowitz, M., and Stegun, I., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Vol. 55, Dover Publ., New York, 1965, pp. 887–889, Chap. 25.

[46] Patterson, M. A., and Rao, A. V., "Exploiting Sparsity in Direct Collocation Pseudospectral Methods for Solving Continuous-Time Optimal Control Problems," *Journal of Spacecraft and Rockets*, Vol. 49, No. 2, March–April 2012, pp. 354–377.
doi:10.2514/1.A32071

[47] Bauer, T., Betts, J., Hallman, W., Huffman, W., and Zondervan, K., "Solving the Optimal Control Problem Using a Nonlinear Programming Technique. II—Optimal Shuttle Ascent Trajectories," *Astrodynamics Conference, Guidance, Navigation, and Control and Co-Located Conferences*, AIAA Paper 1984-2038, 1984.
doi:10.2514/6.1984-2038

[48] Miller, A. T., and Rao, A. V., "Rapid Ascent-Entry Vehicle Mission Optimization Using hp-Adaptive Gaussian Quadrature Collocation," *2017 AIAA Atmospheric Flight Mechanics Conference*, AIAA Paper 2017-0249, Jan. 2017.
doi:10.2514/6.2017-0249