$\mathbb{GPOPS} - \mathbb{II}$: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp–Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming

Michael A. Patterson and Anil V. Rao
University of Florida, Gainesville, FL 32611-6250

A general-purpose MATLAB software program called $\mathbb{GPOPS} - \mathbb{II}$ is described for solving multiple-phase optimal control problems using variable-order Gaussian quadrature collocation methods. The software employs a Legendre-Gauss-Radau quadrature orthogonal collocation method where the continuous-time optimal control problem is transcribed to a large sparse nonlinear programming problem (NLP). An adaptive mesh refinement method is implemented that determines the number of mesh intervals and the degree of the approximating polynomial within each mesh interval to achieve a specified accuracy. The software can be interfaced with either quasi-Newton (first derivative) or Newton (second derivative) NLP solvers, and all derivatives required by the NLP solver are approximated using sparse finite-differencing of the optimal control problem functions. The key components of the software are described in detail and the utility of the software is demonstrated on five optimal control problems of varying complexity. The software described in this paper provides researchers a useful platform upon which to solve a wide variety of complex constrained optimal control problems.

Categories and Subject Descriptors: G.1.4 [**Numerical Analysis**]: Optimal Control, Optimization

General Terms: Optimal Control, Direct Collocation, Gaussian Quadrature, hp–Adaptive Methods, Numerical Methods, MATLAB.

Additional Key Words and Phrases: Scientific Computation, Applied Mathematics.

ACM Reference Format:
Patterson, M. A. and Rao, A. V. 2012. GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp–Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming ACM Trans. Math. Soft. 39, 3, Article 1 (July 2013), 41 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Optimal control problems arise in a wide variety of subjects including virtually all branches of engineering, economics, and medicine. Because optimal control applications have increased in complexity in recent years, over the past two decades the subject of optimal control has transitioned from theory to computation. In particular, computational optimal control has become a science in and of itself, resulting in a variety of numerical methods and corresponding software implementations of these methods. The vast majority of software implementations of optimal control today are

The authors gratefully acknowledge support for this research from the U.S. Office of Naval Research (ONR) under Grant N00014-11-1-0068 and from the U.S. Defense Advanced Research Projects Agency (DARPA) Under Contract HR0011-12-C-0011. **Disclaimer:** The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.
Author's addresses: M. A. Patterson and A. V. Rao, Department of Mechanical and Aerospace Engineering, P.O. Box 116250, University of Florida, Gainesville, FL 32611-6250; e-mail: {mpatterson,anilvrao}@ufl.edu.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

those that involve the direct transcription of a continuous-time optimal control problem to a nonlinear programming problem (NLP), and the NLP is solved using well established techniques. Examples of well known software for solving optimal control problems include *SOCS* [Betts 1998], *DIRCOL* [von Stryk 2000], *GESOP* [Jansch et al. 1994], *OTIS* [Vlases et al. 1990], *MISER* [Goh and Teo 1988], *PSOPT* [Becerra 2009], *GPOPS* [Rao et al. 2010], *ICLOCS* [Falugi et al. 2010], *JModelica* [Åkesson et al. 2010], *ACADO* [Houska et al. 2011], and *Sparse Optimization Suite* (SOS) [Betts 2013].

Over the past two decades, direct collocation methods have become popular in the numerical solution of nonlinear optimal control problems. In a direct collocation method, the state and control are discretized at a set of appropriately chosen points in the time interval of interest. The continuous-time optimal control problem is then transcribed to a finite-dimensional NLP. The resulting NLP is then solved using well known software such as *SNOPT* [Gill et al. 2002], *IPOPT* [Wächter and Biegler 2006; Biegler and Zavala 2008] and *KNITRO* [Byrd et al. 2006]. Originally, direct collocation methods were developed as h methods (for example, Euler or Runge-Kutta methods) where the time interval is divided into a mesh and the state is approximated using the same fixed-degree polynomial in each mesh interval. Convergence in an h method is then achieved by increasing the number and placement of the mesh points [Betts 2010; Jain and Tsiotras 2008; Zhao and Tsiotras 2011]. More recently, a great deal of research as been done in the class of direct *Gaussian quadrature orthogonal collocation* methods [Elnagar et al. 1995; Elnagar and Razzaghi 1998; Benson et al. 2006; Huntington et al. 2007; Huntington and Rao 2008; Gong et al. 2008b; Rao et al. 2010; Garg et al. 2010; Garg et al. 2011a; Garg et al. 2011b; Kameswaran and Biegler 2008; Darby et al. 2011a; Patterson and Rao 2012]. In a Gaussian quadrature collocation method, the state is typically approximated using a Lagrange polynomial where the support points of the Lagrange polynomial are chosen to be points associated with a Gaussian quadrature. Originally, Gaussian quadrature collocation methods were implemented as p methods using a single interval. Convergence of the p method was then achieved by increasing the degree of the polynomial approximation. For problems whose solutions are smooth and well-behaved, a p Gaussian quadrature collocation method has a simple structure and converges at an exponential rate [Canuto et al. 1988; Fornberg 1998; Trefethen 2000]. The most well developed p Gaussian quadrature methods are those that employ either Legendre-Gauss (LG) points [Benson et al. 2006; Rao et al. 2010], Legendre-Gauss-Radau (LGR) points [Kameswaran and Biegler 2008; Garg et al. 2010; Garg et al. 2011a], or Legendre-Gauss-Lobatto (LGL) points [Elnagar et al. 1995].

In this paper we describe a new optimal control software called $\mathbb{GPOPS} - \mathbb{II}$ that employs hp-adaptive Gaussian quadrature collocation methods. An hp-adaptive method is a hybrid between a p method and an h method in that both the number of mesh intervals and the degree of the approximating polynomial within each mesh interval can be varied in order to achieve a specified accuracy in the numerical approximation of the solution to the continuous-time optimal control problem. As a result, in an hp-adaptive method it is possible to take advantage of the exponential convergence of a global Gaussian quadrature method in regions where the solution is smooth and introduce mesh points only near potential discontinuities or in regions where the solution changes rapidly. Originally, hp methods were developed as finite-element methods for solving partial differential equations [Babuska et al. 1986; Babuska and Rheinboldt 1982; 1979; 1981]. In the past few years the problem of developing hp methods for solving optimal control problems has been of interest [Darby et al. 2011b; 2011c]. The work of [Darby et al. 2011b; 2011c] provides examples of the benefits of using an hp–adaptive method over either a p method or an h method. This recent research has

shown that convergence using hp methods can be achieved with a significantly smaller finite-dimensional approximation than would be required when using either an h or a p method.

It is noted that previously the software *GPOPS* was published in Rao et al. [2010]. While *GPOPS* is similar to $\mathbb{GPOPS} - \mathbb{II}$ in that both software programs implement Gaussian quadrature collocation, $\mathbb{GPOPS} - \mathbb{II}$ is a fundamentally different software program from *GPOPS*. First, *GPOPS* employs p (global) collocation in each phase of the optimal control problem. It is known that p collocation schemes are limited in that they have difficulty solving problems whose solutions change rapidly in certain regions or are discontinuous. Moreover, p methods become computationally intractable as the degree of the approximating polynomial becomes very large. $\mathbb{GPOPS} - \mathbb{II}$, however, employs hp-adaptive mesh refinement where the polynomial degree, number of mesh intervals, and width of each mesh interval can be varied. The hp-adaptive methods allow for placement of collocation points in regions of the solution where additional information is needed to capture key features of the optimal solution. Next, *GPOPS* is limited in that it can be used with only quasi-Newton (first derivative) NLP solvers and derivative approximations were performed on high dimensional NLP functions. On the other hand, $\mathbb{GPOPS} - \mathbb{II}$ implements *sparse derivative approximations* by approximating derivatives of the optimal control functions and inserting these derivatives into the appropriate locations in the NLP derivative functions. Moreover, $\mathbb{GPOPS} - \mathbb{II}$ implements approximations to *both* first and second derivatives. Consequently, $\mathbb{GPOPS} - \mathbb{II}$ utilizes in an efficient manner the full capabilities of a much wider range of NLP solvers (for example, full Newton NLP solvers such as *IPOPT* [Biegler and Zavala 2008] and *KNITRO* [Byrd et al. 2006]) and, as a result, is capable of solving a much wider range of optimal control problems as compared with *GPOPS*.

The objective of this paper is to provide researchers with a novel efficient general-purpose optimal control software that is capable of solving a wide variety of complex constrained continuous-time optimal control problems. In particular, the software described in this paper employs a differential and implicit integral form of the multiple-interval version of the *Legendre-Gauss-Radau (LGR) collocation method* [Garg et al. 2010; Garg et al. 2011a; Garg et al. 2011b; Patterson and Rao 2012]. The LGR collocation method is chosen for use in the software because it provides highly accurate state, control, and costate approximations while maintaining a relatively low-dimensional approximation of the continuous-time problem. The key components of the software are then described, and the software is demonstrated on five examples from the open literature that have been studied extensively and whose solutions are known. Each of these examples demonstrates different capabilities of the software. The first example is the hyper-sensitive optimal control problem from Rao and Mease [2000] and demonstrates the ability of the software to accurately solve a problem whose optimal solution changes rapidly in particular regions of the solution. The second example is the reusable launch vehicle entry problem taken from Betts [2010] and demonstrates the ability of $\mathbb{GPOPS} - \mathbb{II}$ to compute an accurate solution using a relatively coarse mesh. The third example is a space station attitude control problem taken from Pietz [2003] and Betts [2010] and demonstrates the ability of the software to generate accurate solutions to a problem whose solution is not intuitive. The fourth example is a chemical kinetic batch reactor problem taken from Leineweber [1998] and Betts [2010] and shows the ability of the software to handle a multiple-phase optimal control problem that is badly scaled. The fifth example is a launch vehicle ascent problem taken from Benson [2004], Rao et al. [2010], and Betts [2010] that again demonstrates the ability of the software to solve a multiple-phase optimal control problem. In order to validate the results, the solutions obtained using $\mathbb{GPOPS} - \mathbb{II}$ are compared against the solu-

tions obtained using the software *Sparse Optimization Suite* (SOS) [Betts 2013] where SOS is based on the collection of algorithms developed in Betts [2010].

This paper is organized as follows. In Section 2 we describe a general multiple-phase optimal control problem. In Section 3 we describe the Radau collocation method that is used as the basis of $\mathbb{GPOPS}-\mathbb{II}$. In Section 4 we describe the key components of $\mathbb{GPOPS}-\mathbb{II}$. In Section 5 we show the results obtained using the software on the five aforementioned examples. In Section 6 we provide a discussion of the results obtained using the software. In Section 7 we discuss possible limitations of the software. Finally, in Section 8 we provide conclusions on our work.

2. GENERAL MULTIPLE-PHASE OPTIMAL CONTROL PROBLEMS

The general multiple-phase optimal control problem that can be solved by $\mathbb{GPOPS}-\mathbb{II}$ is given as follows. First, let $p \in [1,\dots,P]$ be the phase number where P as the total number of phases. The optimal control problem is to determine the state, $\mathbf{y}^{(p)}(t) \in \mathbb{R}^{n_y^{(p)}}$, control, $\mathbf{u}^{(p)}(t) \in \mathbb{R}^{n_u^{(p)}}$, integrals, $\mathbf{q}^{(p)} \in \mathbb{R}^{n_q^{(p)}}$, start times, $t_0^{(p)} \in \mathbb{R}$, phase terminus times, $t_f^{(p)} \in \mathbb{R}$, in all phases $p \in [1,\dots,P]$, along with the static parameters $\mathbf{s} \in \mathbb{R}^{n_s}$, that minimize the objective functional

$$J = \phi\left(\mathbf{e}^{(1)},\dots,\mathbf{e}^{(P)},\mathbf{s}\right), \tag{1}$$

subject to the dynamic constraints

$$\dot{\mathbf{y}}^{(p)} = \mathbf{a}^{(p)}(\mathbf{y}^{(p)},\mathbf{u}^{(p)},t^{(p)},\mathbf{s}), \quad (p=1,\dots,P), \tag{2}$$

the event constraints

$$\mathbf{b}_{\min} \le \mathbf{b}\left(\mathbf{e}^{(1)},\dots,\mathbf{e}^{(P)},\mathbf{s}\right) \le \mathbf{b}_{\max}, \tag{3}$$

where $\mathbf{e}^{(p)} = \left[\mathbf{y}^{(p)}(t_0^{(p)}),t_0^{(p)},\mathbf{y}^{(p)}(t_f^{(p)}),t_f^{(p)},\mathbf{q}^{(p)}\right]$, $(p=1,\dots,P)$, the inequality path constraints

$$\mathbf{c}_{\min}^{(p)} \le \mathbf{c}^{(p)}(\mathbf{y}^{(p)},\mathbf{u}^{(p)},t^{(p)},\mathbf{s}) \le \mathbf{c}_{\max}^{(p)}, \quad (p=1,\dots,P), \tag{4}$$

the static parameter constraints

$$\mathbf{s}_{\min} \le \mathbf{s} \le \mathbf{s}_{\max}, \tag{5}$$

and the integral constraints

$$\mathbf{q}_{\min}^{(p)} \le \mathbf{q}^{(p)} \le \mathbf{q}_{\max}^{(p)}, \quad (p=1,\dots,P), \tag{6}$$

where

$$\mathbf{e}^{(p)} = \left[\mathbf{y}^{(p)}\left(t_0^{(p)}\right),t_0^{(p)},\mathbf{y}^{(p)}\left(t_f^{(p)}\right),t_f^{(p)},\mathbf{q}^{(p)}\right], \ (p=1,\dots,P), \tag{7}$$

and the integrals in each phase are defined as

$$q_i^{(p)} = \int_{t_0^{(p)}}^{t_f^{(p)}} g_i^{(p)}(\mathbf{y}^{(p)},\mathbf{u}^{(p)},t^{(p)},\mathbf{s})dt, \quad (i=1,\dots n_q^{(p)}),(p=1,\dots,P). \tag{8}$$

It is important to note that the event constraints of Eq. (3) can contain any functions that relate information at the start and/or terminus of any phase (including relationships that include both static parameters and integrals) and that the phases themselves need not be sequential. It is noted that the approach to linking phases is based on well-known formulations in the literature such as those given in Ref. Betts [1990] and Betts [2010]. A schematic of how phases can potentially be linked is given in Fig. 1.

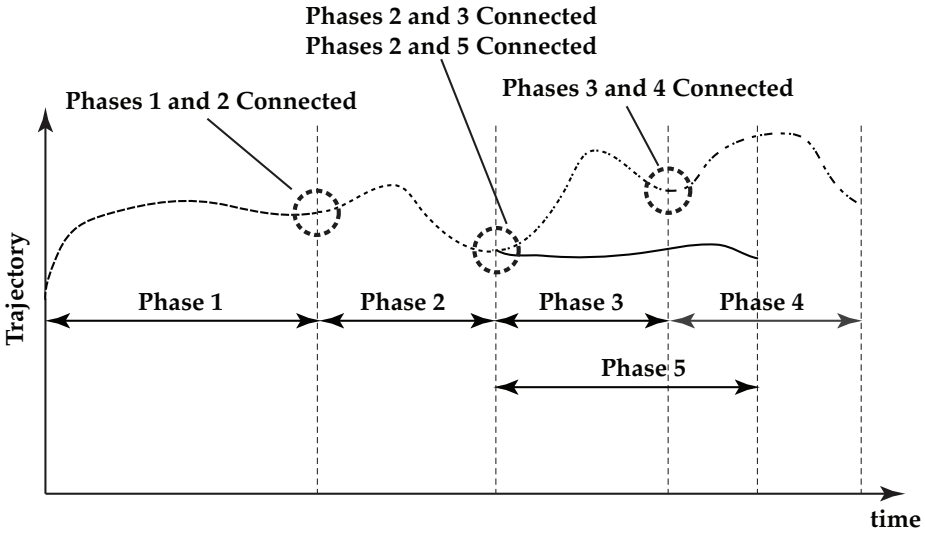


Fig. 1: Schematic of linkages for multiple-phase optimal control problem. The example shown in the picture consists of five phases where the ends of phases 1, 2, and 3 are linked to the starts of phases 2, 3, and 4, respectively, while the end of phase 2 is linked to the start of phase 5.

3. MULTIPLE-INTERVAL RADAU COLLOCATION METHOD

In this section we describe the multiple-interval Radau collocation method [Garg et al. 2010; Garg et al. 2011a; Garg et al. 2011b; Patterson and Rao 2012] that forms the basis for $\mathbb{GPOPS}-\mathbb{II}$. In order to make the description of the Radau collocation method as clear as possible, in this section we consider only a one-phase optimal control problem. After formulating the one-phase optimal control problem we develop the Radau collocation method itself.

3.1. Single-Phase Optimal Control Problem

In order to describe the hp Radau method that is implemented in the software, it will be useful to simplify the general optimal control problem given in Eqs. (1)–(8) to a one-phase problem as follows. Determine the state, $\mathbf{y}(t) \in \mathbb{R}^{n_y}$, the control, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$, the integrals, $\mathbf{q} \in \mathbb{R}^{n_q}$, the initial time, t_0, and the terminal time t_f on the time interval $t \in [t_0, t_f]$ that minimize the cost functional

$$\mathcal{J} = \phi(\mathbf{y}(t_0), t_0, \mathbf{y}(t_f), t_f, \mathbf{q}) \tag{9}$$

subject to the dynamic constraints

$$\frac{d\mathbf{y}}{dt} = \mathbf{a}(\mathbf{y}(t), \mathbf{u}(t), t), \tag{10}$$

the inequality path constraints

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{y}(t), \mathbf{u}(t), t) \leq \mathbf{c}_{\max}, \tag{11}$$

the integral constraints

$$q_i = \int_{t_0}^{t_f} g_i(\mathbf{y}(t), \mathbf{u}(t), t)\, dt, \quad (i = 1, \ldots, n_q), \tag{12}$$

and the event constraints

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{y}(t_0), t_0, \mathbf{y}(t_f), t_f, \mathbf{q}) \leq \mathbf{b}_{\min}. \tag{13}$$

The functions ϕ, $\mathbf{q}$, $\mathbf{a}$, $\mathbf{c}$ and $\mathbf{b}$ are defined by the following mappings:

$$\begin{array}{rl} \phi & : \mathbb{R}^{n_y} \times \mathbb{R} \times \mathbb{R}^{n_y} \times \mathbb{R} \times \mathbb{R}^{n_q} \rightarrow \mathbb{R}, \\ \mathbf{q} & : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \rightarrow \mathbb{R}^{n_q}, \\ \mathbf{a} & : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \rightarrow \mathbb{R}^{n_y}, \\ \mathbf{c} & : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \rightarrow \mathbb{R}^{n_c}, \\ \mathbf{b} & : \mathbb{R}^{n_y} \times \mathbb{R} \times \mathbb{R}^{n_y} \times \mathbb{R} \times \mathbb{R}^{n_q} \rightarrow \mathbb{R}^{n_b}, \end{array}$$

where we remind the reader that all vector functions of time are treated as *row* vectors.

In order to employ the hp Radau collocation method used in $\mathbb{GPOPS} - \mathbb{II}$, the continuous optimal control problem of Eqs. (9)–(13) is modified as follows. First, let $\tau \in [-1, +1]$ be a new independent variable. The variable t is then defined in terms of τ as

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}. \tag{14}$$

The optimal control problem of Eqs. (9)–(13) is then defined in terms of the variable τ as follows. Determine the state, $\mathbf{y}(\tau) \in \mathbb{R}^{n_y}$, the control $\mathbf{u}(\tau) \in \mathbb{R}^{n_u}$, the integral $\mathbf{q} \in \mathbb{R}^{n_q}$, the initial time, t_0, and the terminal time t_f on the time interval $\tau \in [-1, +1]$ that minimize the cost functional

$$\mathcal{J} = \phi(\mathbf{y}(-1), t_0, \mathbf{y}(+1), t_f, \mathbf{q}) \tag{15}$$

subject to the dynamic constraints

$$\frac{d\mathbf{y}}{d\tau} = \frac{t_f - t_0}{2}\mathbf{a}(\mathbf{y}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f), \tag{16}$$

the inequality path constraints

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{y}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \leq \mathbf{c}_{\max}, \tag{17}$$

the integral constraints

$$q_i = \frac{t_f - t_0}{2} \int_{-1}^{+1} g_i(\mathbf{y}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f)\, d\tau, \quad (i = 1, \ldots, n_q), \tag{18}$$

and the event constraints

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{y}(-1), t_0, \mathbf{y}(+1), t_f, \mathbf{q}) \leq \mathbf{b}_{\min}. \tag{19}$$

Suppose now that the interval $\tau \in [-1, +1]$ is divided into a *mesh* consisting of K *mesh intervals* $[T_{k-1}, T_k]$, $k = 1, \ldots, K$, where $(T_0, \ldots, T_K)$ are the *mesh points*. The mesh points have the property that $-1 = T_0 < T_1 < T_2 < \cdots < T_K = T_f = +1$. Next, let $\mathbf{y}^{(k)}(\tau)$ and $\mathbf{u}^{(k)}(\tau)$ be the state and control in mesh interval k. The optimal control problem of Eqs. (15)–(19) can then written as follows. First, the cost functional of Eq. (15) can be written as

$$\mathcal{J} = \phi(\mathbf{y}^{(1)}(-1), t_0, \mathbf{y}^{(K)}(+1), t_f, \mathbf{q}), \tag{20}$$

Next, the dynamic constraints of Eq. (16) in mesh interval k can be written as

$$\frac{d\mathbf{y}^{(k)}(\tau^{(k)})}{d\tau^{(k)}} = \frac{t_f - t_0}{2}\mathbf{a}(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f), \quad (k = 1, \ldots, K). \tag{21}$$

Furthermore, the path constraints of (17) in mesh interval k are given as

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f) \leq \mathbf{c}_{\max}, \quad (k = 1, \ldots, K). \tag{22}$$

the integral constraints of (18) are given as

$$q_j = \frac{t_f - t_0}{2} \sum_{k=1}^{K} \int_{T_{k-1}}^{T_k} g_j(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f)\, d\tau, \quad (j = 1, \ldots, n_q, k = 1 \ldots, K). \tag{23}$$

Finally, the event constraints of Eq. (19) are given as

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{y}^{(1)}(-1), t_0, \mathbf{y}^{(K)}(+1), t_f, \mathbf{q}) \leq \mathbf{b}_{\max}. \tag{24}$$

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

3.2. Approximation of the Optimal Control Problem via Radau Collocation Method

The method utilized in the software is an implementation of the previously developed multiple-interval Legendre-Gauss-Radau quadrature orthogonal collocation method (known hereafter as the *Radau collocation method*) [Garg et al. 2010; Garg et al. 2011a; Garg et al. 2011b; Patterson and Rao 2012]. In the Radau collocation method, the state of the continuous-time optimal control problem is approximated in each mesh interval $k \in [1, \ldots, K]$ as

$$\mathbf{y}^{(k)}(\tau) \approx \mathbf{Y}^{(k)}(\tau) = \sum_{j=1}^{N_k+1} \mathbf{Y}_j^{(k)} \ell_j^{(k)}(\tau), \quad \ell_j^{(k)}(\tau) = \prod_{\substack{l=1 \\ l \neq j}}^{N_k+1} \frac{\tau - \tau_l^{(k)}}{\tau_j^{(k)} - \tau_l^{(k)}}, \tag{25}$$

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 Legendre-Gauss-Radau [Abramowitz and Stegun 1965] (LGR) collocation points in mesh interval k defined on the subinterval $\tau^{(k)} \in [T_{k-1}, T_k)$, and $\tau_{N_k+1}^{(k)} = T_k$ is a noncollocated point. Differentiating $\mathbf{Y}^{(k)}(\tau)$ in Eq. (25) with respect to τ, we obtain

$$\frac{d\mathbf{Y}^{(k)}(\tau)}{d\tau} = \sum_{j=1}^{N_k+1} \mathbf{Y}_j^{(k)} \frac{d\ell_j^{(k)}(\tau)}{d\tau}. \tag{26}$$

The cost functional of Eq. (20) is then shown as

$$\mathcal{J} = \phi(\mathbf{Y}_1^{(1)}, t_0, \mathbf{Y}_{N_K+1}^{(K)}, t_f, \mathbf{q}), \tag{27}$$

where $\mathbf{Y}_1^{(1)}$ is the approximation of $\mathbf{y}(T_0 = -1)$, and $\mathbf{Y}_{N_K+1}^{(K)}$ is the approximation of $\mathbf{y}(T_K = +1)$. Collocating the dynamics of Eq. (21) at the N_k LGR points using Eq. (26), we have

$$\sum_{j=1}^{N_k+1} D_{ij}^{(k)} \mathbf{Y}_j^{(k)} - \frac{t_f - t_0}{2} \mathbf{a}(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) = \mathbf{0}, \quad (i = 1, \ldots, N_k). \tag{28}$$

where $\mathbf{U}_i^{(k)}$, $i = 1, \ldots, N_k$, are the approximations of the control at the N_k LGR points in mesh interval $k \in [1, \ldots, K]$, and $t_i^{(k)}$ are obtained from $\tau_k^{(k)}$ using Eq. (14) and

$$D_{ij}^{(k)} = \left[\frac{d\ell_j^{(k)}(\tau)}{d\tau} \right]_{\tau_i^{(k)}}, \quad (i = 1, \ldots, N_k, \quad j = 1, \ldots, N_k + 1, \quad k = 1, \ldots, K), \tag{29}$$

is the $N_k \times (N_k+1)$ *Legendre-Gauss-Radau differentiation matrix* [Garg et al. 2010] in mesh interval $k \in [1,\ldots,K]$. While the dynamics can be collocated in differential form, an alternative approach is to collocate the dynamics using the equivalent *implicit integral form* of the Radau collocation method as described in Garg et al. [2010; Garg et al. [2011a; Garg et al. [2011b]. Collocating the dynamics using the implicit integral form of the Radau collocation method we have

$$\mathbf{Y}_{i+1}^{(k)} - \mathbf{Y}_1^{(k)} - \frac{t_f - t_0}{2}\sum_{j=1}^{N_k} I_{ij}^{(k)}\mathbf{a}(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) = \mathbf{0}, \quad (i = 1,\ldots,N_k), \tag{30}$$

where $I_{ij}^{(k)},\ (i = 1,\ldots,N_k, j = 1,\ldots,N_k, k = 1,\ldots,K)$ is the $N_k \times N_k$ *Legendre-Gauss-Radau integration matrix* in mesh interval $k \in [1,\ldots,K]$, and is obtained from the differentiation matrix as [Garg et al. 2010; Garg et al. 2011a; Garg et al. 2011b]

$$\mathbf{I}^{(k)} \equiv \left[\mathbf{D}_{2:N_k+1}^{(k)}\right]^{-1},$$

Finally, it is noted for completeness that $\mathbf{I}^{(k)}\mathbf{D}_1^{(k)} = -\mathbf{1}$, where $\mathbf{1}$ is a column vector of length N_k of all ones. It is noted that Eqs. (28) and (30) can be be evaluated over all intervals simultaneously using the composite Legendre-Gauss-Radau differentiation matrix $\mathbf{D}$, and the composite Legendre-Gauss-Radau integration matrix $\mathbf{I}$ respectively. Furthermore, the sparse structure of the composite Legendre-Gauss-Radau differentiation matrix $\mathbf{D}$ can be seen in Fig. 2, and the structure of the composite Legendre-Gauss-Radau integration matrix $\mathbf{I}$ can be seen in Fig. 3. Next, the path constraints of Eq. (22) in mesh interval $k \in [1,\ldots,K]$ are enforced at the N_k LGR points as

$$\mathbf{c}_{\min} \le \mathbf{c}(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) \le \mathbf{c}_{\max}, \quad (i = 1,\ldots,N_k), \tag{31}$$

the integral constraints of Eq. (23) is then approximated as

$$q_j \approx \sum_{k=1}^{K}\sum_{i=1}^{N_k} \frac{t_f - t_0}{2} w_i^{(k)} g_j(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f), \quad (i = 1,\ldots,N_k, j = 1,\ldots,n_q), \tag{32}$$

where $w_j^{(k)},\ j = 1,\ldots,N_k$ are the LGR quadrature weights [Abramowitz and Stegun 1965] in mesh interval $k \in [1,\ldots,K]$ defined on the interval $\tau \in [\tau_{k-1}, \tau_k)$. Furthermore, the event constraints of Eq. (24) are approximated as

$$\mathbf{b}_{\min} \le \mathbf{b}(\mathbf{Y}_1^{(1)}, t_0, \mathbf{Y}_{N_K+1}^{(K)}, t_f, \mathbf{q}) \le \mathbf{b}_{\max}. \tag{33}$$

It is noted that continuity in the state at the interior mesh points $k \in [1,\ldots,K-1]$ is enforced via the condition

$$\mathbf{Y}_{N_k+1}^{(k)} = \mathbf{Y}_1^{(k+1)}, \quad (k = 1,\ldots,K-1), \tag{34}$$

where we note that the *same* variable is used for both $\mathbf{Y}_{N_k+1}^{(k)}$ and $\mathbf{Y}_1^{(k+1)}$ in the software implementation. Hence, the constraint of Eq. (34) is eliminated from the problem because it is taken into account explicitly. The NLP that arises from the Radau collocation method is then to minimize the cost function of Eq. (27) subject to the algebraic constraints of Eqs. (28)–(33).

4. MAJOR COMPONENTS OF $\mathbb{GPOPS}-\mathbb{II}$

In this section we describe the major components of the MATLAB software $\mathbb{GPOPS}-\mathbb{II}$ that implements the aforementioned Radau collocation method. In Section 4.1 we describe the large sparse nonlinear programming problem (NLP) associated with the

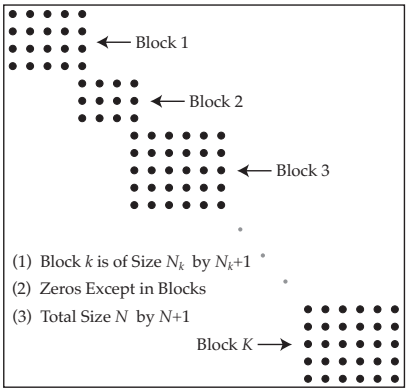


Fig. 2: Structure of Composite Legendre-Gauss-Radau Differentiation Matrix Where the Mesh Consists of K Mesh Intervals.

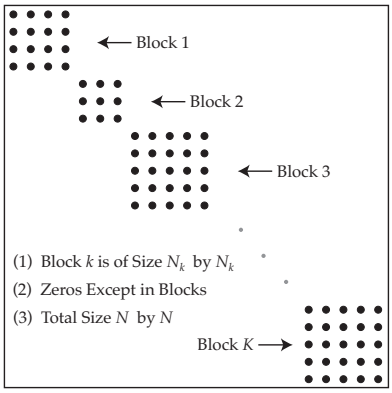


Fig. 3: Structure of Composite Legendre-Gauss-Radau Integration Matrix Where the Mesh Consists of K Mesh Intervals.

Radau collocation method. In Section 4.2 we show the structure of the NLP described in Section 4.1. In Section 4.3 we describe the method for scaling the NLP via scaling of the optimal control problem. In Section 4.4 we describe the approach for estimating the derivatives required by the NLP solver. In Section 4.5 we describe the method for determining the dependencies of each optimal control function in order to provide the most sparse NLP to the NLP solver. In Section 4.6 we describe the hp-adaptive mesh refinement methods that are included in the software in order to iterative determine a mesh that meets a user-specified accuracy tolerance. Finally, in Section 4.7 we provide a high level description of the algorithmic flow of $\mathbb{GPOPS} - \mathbb{II}$.

4.1. Sparse NLP Arising from Multiple-Interval Radau Collocation Method

The nonlinear programming problem (NLP) associated with the hp Radau discretized scaled continuous-time optimal control problem is given as follows. Determine the decision vector $\mathbf{Z}$ that minimizes the objective function

$$\Phi(\mathbf{Z}) \tag{35}$$

subject to the constraints

$$\mathbf{F}_{\min} \leq \mathbf{F}(\mathbf{Z}) \leq \mathbf{F}_{\max}. \tag{36}$$

and the variable bounds

$$\mathbf{Z}_{\min} \leq \mathbf{Z} \leq \mathbf{Z}_{\max}. \tag{37}$$

It is noted that the size of the NLP arising from the hp Radau collocation method changes depending upon the number of mesh intervals and LGR points used in each phase to discretize the continuous-time optimal control problem, but the structure of the NLP is the same regardless of the number of mesh intervals or number of LGR points used in the discretization.

4.1.1. NLP Variables. The NLP decision vector $\mathbf{Z}$ is given as

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(P)} \\ s_1 \\ \vdots \\ s_{n_s} \end{bmatrix}, \tag{38}$$

where $\mathbf{z}^{(p)}$ contains all the variables of phase $p = 1 \dots P$, and s_i $(i = 1, \dots, n_s)$ are the static parameters in the problem. The phase-dependent variables of Eq. (38) $\mathbf{z}^{(p)}$ $(p = 1, \dots, P)$, are given as

$$\mathbf{z}^{(p)} = \begin{bmatrix} \mathbf{V}_1^{(p)} \\ \vdots \\ \mathbf{V}_{n_y^{(p)}}^{(p)} \\ \mathbf{W}_1^{(p)} \\ \vdots \\ \mathbf{W}_{n_u^{(p)}}^{(p)} \\ \mathbf{q}^{(p)} \\ t_0^{(p)} \\ t_f^{(p)} \end{bmatrix}, \quad (p = 1, \dots, P), \tag{39}$$

where $\mathbf{V}_i^{(p)} \in \mathbb{R}^{(N^{(p)}+1)}$ $(i = 1, \dots, n_y^{(p)})$ is the i^{th} column of the matrix

$$\mathbf{V}^{(p)} = \begin{bmatrix} \mathbf{Y}_1^{(p)} \\ \vdots \\ \mathbf{Y}_{N^{(p)}+1}^{(p)} \end{bmatrix} \in \mathbb{R}^{(N^{(p)}+1) \times n_y^{(p)}}, \tag{40}$$

$\mathbf{W}_i^{(p)} \in \mathbb{R}^{N^{(p)}}$ $(i = 1, \dots, n_u^{(p)})$ is the i^{th} column of the matrix,

$$\mathbf{W}^{(p)} = \begin{bmatrix} \mathbf{U}_1^{(p)} \\ \vdots \\ \mathbf{U}_{N^{(p)}}^{(p)} \end{bmatrix} \in \mathbb{R}^{N^{(p)} \times n_u^{(p)}}, \tag{41}$$

$\mathbf{q}^{(p)}$ is a column vector containing the $n_q^{(p)}$ integral constraint variables, and $t_0^{(p)} \in \mathbb{R}$ and $t_f^{(p)} \in \mathbb{R}$ are scalars corresponding to the initial and terminal times in phase $p \in [1,\ldots,P]$. We remind the reader again that because the state and control are being treated as *row* vectors, the i^{th} row in the matrices of Eqs. (40) and (40) corresponds to the value of the discretized state and control at the time $\tau_i^{(p)}$. Finally, the bounds $\mathbf{Z}_{\min}$ and $\mathbf{Z}_{\max}$ are defined from the optimal control variable bounds as supplied by the user.

4.1.2. NLP Objective and Constraint Functions. The NLP objective function $\Phi(\mathbf{Z})$ is defined as follows

$$\Phi(\mathbf{Z}) = \phi \tag{42}$$

where ϕ is the optimal control objective function evaluated at the discrete variables defined in function of Eq. (1). The NLP constraint function vector $\mathbf{F}(\mathbf{Z})$ is then assembled as

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}^{(1)} \\ \vdots \\ \mathbf{f}^{(P)} \\ \mathbf{b} \end{bmatrix}, \tag{43}$$

where $\mathbf{f}^{(p)}$ are the constraints in phase $p \in [1,\ldots,P]$ and $\mathbf{b}$ is the vector of $n_b^{(p)}$ event constraints evaluated at the discrete variables defined in function of Eq. (3). The phase-dependent constraints of Eq. (43) $\mathbf{f}^{(p)}$ $(p \in [1,\ldots,P]$ have the structure

$$\mathbf{f}^{(p)} = \begin{bmatrix} \boldsymbol{\Delta}_1^{(p)} \\ \vdots \\ \boldsymbol{\Delta}_{n_y^{(p)}}^{(p)} \\ \mathbf{C}_1^{(p)} \\ \vdots \\ \mathbf{C}_{n_c^{(p)}}^{(p)} \\ \boldsymbol{\rho}^{(p)} \end{bmatrix}, \quad (p = 1,\ldots,P), \tag{44}$$

where $\boldsymbol{\Delta}_i^{(p)} \in \mathbb{R}^{N^{(p)}}$ $(i = 1,\ldots,n_y^{(p)})$ is the i^{th} column in the defect constraint matrix that results from either the differential or implicit integral form of the Radau collocation method. The defect matrix that results from the differential form is defined as

$$\boldsymbol{\Delta}^{(p)} = \mathbf{D}^{(p)}\mathbf{Y}^{(p)} - \frac{t_f^{(p)} - t_0^{(p)}}{2}\mathbf{A}^{(p)} \in \mathbb{R}^{N^{(p)}\times n_y^{(p)}}. \tag{45}$$

The defect matrix that results from the implicit integral form is defined as

$$\boldsymbol{\Delta}^{(p)} = \mathbf{E}^{(p)}\mathbf{Y}^{(p)} - \frac{t_f^{(p)} - t_0^{(p)}}{2}\mathbf{I}^{(p)}\mathbf{A}^{(p)} \in \mathbb{R}^{N^{(p)}\times n_y^{(p)}}, \tag{46}$$

where the matrix $\mathbf{A}$ is the right hand side of the dynamics evaluated at each collocation point

$$\mathbf{A}^{(p)} = \begin{bmatrix} \mathbf{a}(\mathbf{Y}_1^{(p)}, \mathbf{U}_1^{(p)}, t_1^{(p)}, \mathbf{s}) \\ \vdots \\ \mathbf{a}(\mathbf{Y}_{N^{(p)}}^{(p)}, \mathbf{U}_{N^{(p)}}^{(p)}, t_{N^{(p)}}^{(p)}, \mathbf{s}) \end{bmatrix} \in \mathbb{R}^{N^{(p)}\times n_y^{(p)}}, \tag{47}$$

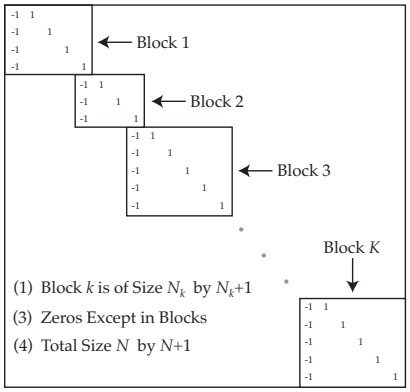


Fig. 4: Matrix $\mathbf{E}$ Corresponding to the Defect Constraints of Eq. (46) for One Phase of a P-Phase Optimal Control Problem Discretized Using the Radau Collocation Method.

$\mathbf{E}^{(p)}$ is an $N^{(p)} \times (N^{(p)}+1)$ matrix that is used to compute the difference $\mathbf{Y}_{i+1}^{(p,k)} - \mathbf{Y}_1^{(p,k)}$ for each interval $k = 1, \ldots, K^{(p)}$, and each phase $p = 1, \ldots, P$ and has the structure shown in Fig. 4, $\mathbf{C}_i^{(p)} \in \mathbb{R}^{N^{(p)}}$, $(i = 1, \ldots, n_c^{(p)})$, is the i^{th} column of the path constraint matrix

$$\mathbf{C}^{(p)} = \begin{bmatrix} \mathbf{c}(\mathbf{Y}_1^{(p)}, \mathbf{U}_1^{(p)}, t_1^{(p)}, \mathbf{s}) \\ \vdots \\ \mathbf{c}(\mathbf{Y}_{N^{(p)}}^{(p)}, \mathbf{U}_{N^{(p)}}^{(p)}, t_{N^{(p)}}^{(p)}, \mathbf{s}) \end{bmatrix} \in \mathbb{R}^{N^{(p)} \times n_c^{(p)}}, \tag{48}$$

$\boldsymbol{\rho}^{(p)}$ is a vector of length $n_{q^{(p)}}^{(p)}$ where the i^{th} element of $\boldsymbol{\rho}^{(p)}$ is given as

$$\rho_i^{(p)} = q_i^{(p)} - \frac{t_f^{(p)} - t_0^{(p)}}{2} \left[\mathbf{w}^{(p)}\right]^{\mathsf{T}} \mathbf{G}_i^{(p)}, \quad (i = 1, \ldots, n_q^{(p)}), \tag{49}$$

where $\mathbf{G}_i^{(p)} \in \mathbb{R}^{N^{(p)}}$ $(i = 1, \ldots, n_q^{(p)})$ is the i^{th} column of the integrand matrix

$$\mathbf{G}^{(p)} = \begin{bmatrix} \mathbf{g}(\mathbf{Y}_1^{(p)}, \mathbf{U}_1^{(p)}, t_1^{(p)}, \mathbf{s}) \\ \vdots \\ \mathbf{g}(\mathbf{Y}_{N^{(p)}}^{(p)}, \mathbf{U}_{N^{(p)}}^{(p)}, t_{N^{(p)}}^{(p)}, \mathbf{s}) \end{bmatrix} \in \mathbb{R}^{N^{(p)} \times n_q^{(p)}}. \tag{50}$$

It is noted that the defect and integral constraints are all equality constraints of the form

$$\begin{aligned} \boldsymbol{\Delta}^{(p)} &= \mathbf{0}, \\ \boldsymbol{\rho}^{(p)} &= \mathbf{0}, \end{aligned} \tag{51}$$

while the discretized path constraints and boundary conditions are inequality constraints of the form

$$\begin{aligned} \mathbf{C}_{\min}^{(p)} \leq \mathbf{C}^{(p)} &\leq \mathbf{C}_{\max}^{(p)}, \\ \mathbf{b}_{\min} \leq \mathbf{b} &\leq \mathbf{b}_{\max}. \end{aligned} \tag{52}$$

4.2. Sparse Structure of NLP Derivative Functions

The structure of the NLP created by the Radau collocation method has been described in detail in Patterson and Rao [2012]. Figures 5 and 6 show the structure of the NLP

constraint Jacobian and Lagrangian Hessian for a single-phase optimal control problem using the differential and integral forms of the Radau method. For a multiple-phase optimal control problem, the constraint Jacobian structure in either the differential or integral form is a block-diagonal version of the single-phase NLP constraint Jacobian where, excluding the event constraints and the static parameters, each block has the structure as seen in either Fig. 5a or 6a. The derivatives of the event constraints are located in the final rows of the constraint Jacobian, while the derivatives of all functions with respect to the static parameters are located in the final columns of the constraint Jacobian. Similarly, the multiple-phase NLP Lagrangian Hessian using either the differential or integral form of the Radau method consists of a block-diagonal version of the single-phase NLP Lagrangian Hessian where, excluding the static parameters, each block has the structure seen in either Fig. 5b or 6b, with additional nonzero elements arising from the second derivatives of the event constraints, b, and the objective function, ϕ. The second derivatives of the Lagrangian with respect to the static parameters appear in the final rows and columns of the Lagrangian Hessian. For more details details on the sparse structure of the NLP arising from the multiple-phase Radau collocation method, please see Patterson and Rao [2012].

4.3. Optimal Control Problem Scaling for NLP

The NLP described in Section 4.1 must be well scaled (that is, the variables and constraints should each be as close as possible $\mathcal{O}(1)$ as described in Gill et al. [1981]) in order for the NLP solver to obtain a solution. $\mathbb{GPOPS}-\mathbb{II}$ includes the option for the NLP to be scaled automatically by scaling the continuous-time optimal control problem. The approach to automatic scaling is to scale the variables and the first derivatives of the optimal control functions to be $\approx \mathcal{O}(1)$. First, the optimal control variables are scaled to lie on the unit interval $[-1/2, 1/2]$ and is accomplished as follows. Suppose it is desired to scale an arbitrary variable $x \in [a, b]$ to $\tilde{x}$ such that $\tilde{x} \in [-1/2, 1/2]$. This variable scaling is accomplished via the affine transformation

$$\tilde{x} = v_x x + r_x, \tag{53}$$

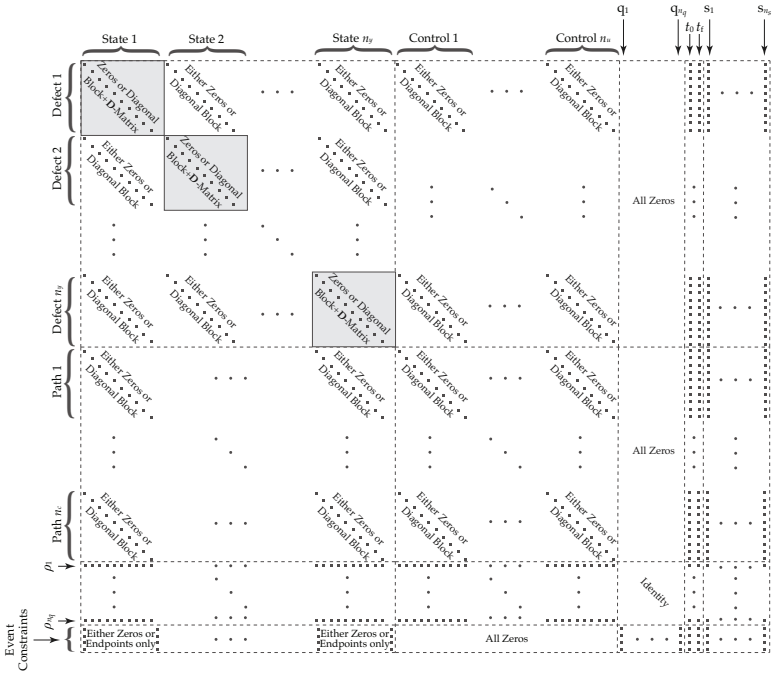
where v_x and r_x are the variable scale and shift, respectively, defined as

$$\begin{aligned} v_x &= \frac{1}{b-a}, \\ r_x &= \frac{1}{2} - \frac{b}{b-a}. \end{aligned} \tag{54}$$

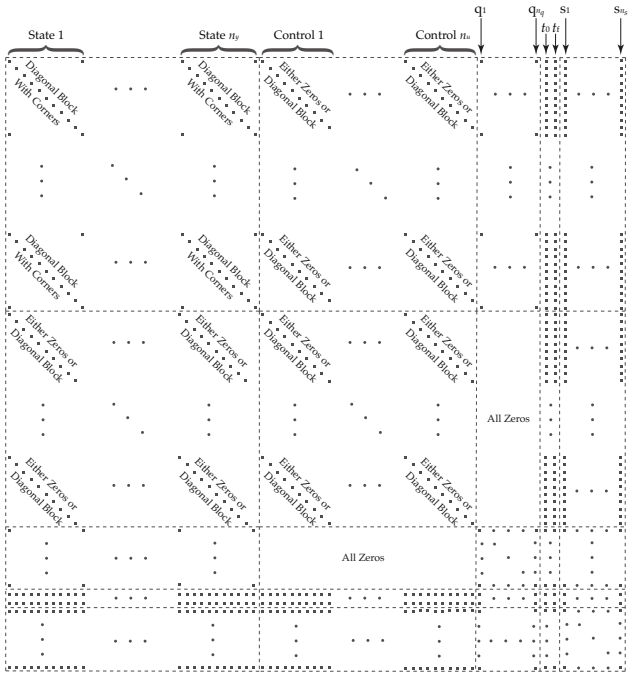
Every variable in the continuous-time optimal control problem is scaled using Eqs. (53) and (54). Next, the Jacobian of the NLP constraints can be made approximately $\mathcal{O}(1)$ by scaling the derivatives of the optimal control functions to be approximately unity. First, using the approach derived in [Betts 2010], in $\mathbb{GPOPS}-\mathbb{II}$ the defect constraints are scaled using the same scale factors as was used to scale the state. Next, the objective function, event constraint, and path constraint scale factors are obtained as the average norm of each gradient across a variety of sample points within the bounds of the unscaled optimal control problem, where the scale factor is the reciprocal of the appropriate average norm.

4.4. Computation of Derivatives Required by the NLP Solver

The optimal control problem derivative functions are obtained by exploiting the sparse structure of the NLP arising from the hp Radau collocation method. Specifically, in Patterson and Rao [2012] it has been shown that using either the derivative or integral form of the Radau collocation method the NLP derivatives can be obtained by computing the derivatives of the optimal control problem functions at the LGR points and

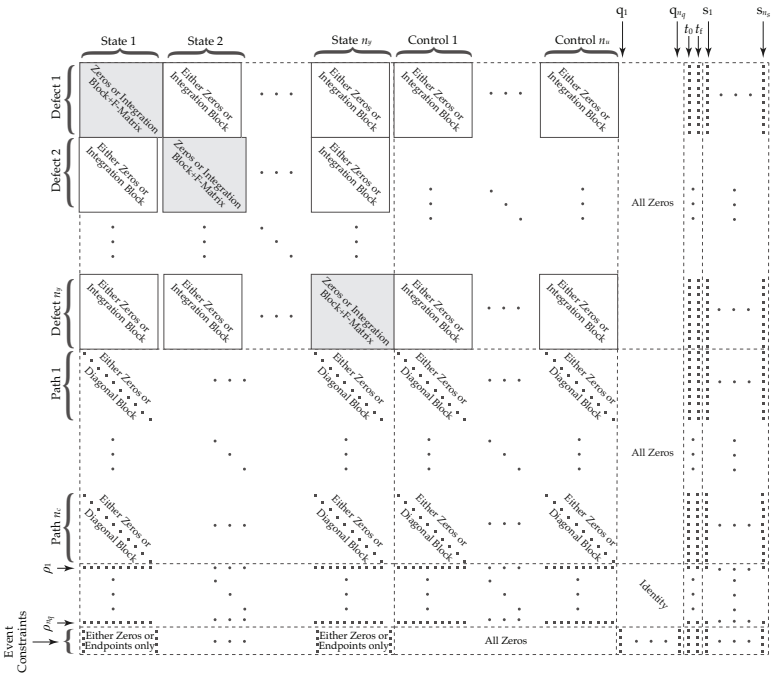


(a) One-Phase NLP Jacobian.

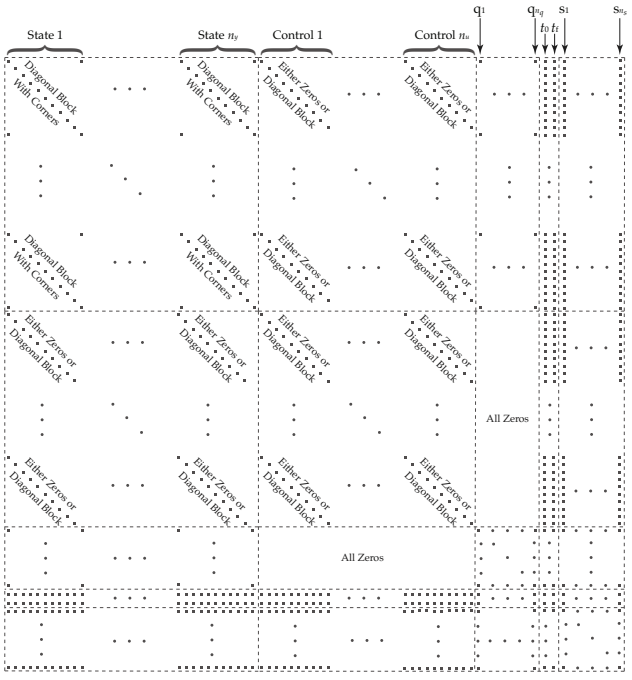


(b) One-Phase NLP Lagrangian Hessian.

Fig. 5: One-Phase Differential Radau Collocation NLP Constraint Jacobian and Lagrangian Hessian.



(a) One-Phase NLP Jacobian.



(b) One-Phase NLP Lagrangian Hessian.

Fig. 6: One-Phase Integral Radau Collocation Method NLP Constraint Jacobian and Lagrangian Hessian.

inserting these derivatives into the appropriate locations in the NLP derivative functions. In $\mathbb{GPOPS}-\mathbb{II}$ the optimal control derivative functions are approximated using sparse forward, central, or backward finite-differencing of the optimal control problem functions. To see how this sparse finite differencing works in practice, consider the function $\mathbf{f}(\mathbf{x})$, where $\mathbf{f}:\mathbb{R}^n \longrightarrow \mathbb{R}^m$ is one of the *optimal control functions* (that is, n and m are, respectively, the size of an optimal control variable and an optimal control function). Then $\partial \mathbf{f}/\partial \mathbf{x}$ is approximated using a forward finite difference as

$$\frac{\partial \mathbf{f}}{\partial x_i} \approx \frac{\mathbf{f}(\mathbf{x}+\mathbf{h}_i)-\mathbf{f}(\mathbf{x})}{h_i}, \tag{55}$$

where $\mathbf{h}_i$ arises from perturbing the i^{th} component of $\mathbf{x}$. The vector $\mathbf{h}_i$ is computed as

$$\mathbf{h}_i = h_i \mathbf{e}_i \tag{56}$$

where $\mathbf{e}_i$ is the i^{th} row of the $n \times n$ identity matrix and h_i is the perturbation size associated with x_i. The perturbation h_i is computed using the equation

$$h_i = h(1+|x_i|), \tag{57}$$

where the base perturbation size h is chosen to be the optimal step size for a function whose input and output are $\approx \mathcal{O}(1)$ as described in Gill et al. [1981]. Second derivative approximations are computed in a manner similar to that used for first derivative approximations with the key difference being that perturbations in two variables are performed. For example, $\partial^2 \mathbf{f}/\partial x_i \partial x_j$ can be approximated using a second forward difference approximation as

$$\frac{\partial^2 \mathbf{f}}{\partial x_i \partial x_j} \approx \frac{\mathbf{f}(\mathbf{x}+\mathbf{h}_i+\mathbf{h}_j)+\mathbf{f}(\mathbf{x})-\mathbf{f}(\mathbf{x}+\mathbf{h}_i)-\mathbf{f}(\mathbf{x}+\mathbf{h}_j)}{h_i h_j}, \tag{58}$$

where $\mathbf{h}_i$, $\mathbf{h}_j$, h_i, and h_j are as defined in Eqs. (56) and (57). The base perturbation size is chosen to minimize round-off error in the finite difference approximation. Furthermore, it is noted that $h_i \longrightarrow h$ as $|x_i| \longrightarrow 0$.

4.5. Method for Determining the Optimal Control Function Dependencies

It can be seen from Section 4.2 that the NLP associated with the Radau collocation method has a sparse structure where the blocks of the constraint Jacobian and Lagrangian Hessian are dependent upon whether a particular NLP function depends upon a particular NLP variable as was shown in Patterson and Rao [2012]. The method for determining the optimal control function dependencies in $\mathbb{GPOPS}-\mathbb{II}$ utilizes the IEEE arithmetic representation of "not-a-number" (NaN) in MATLAB. Specifically, MATLAB has the feature that any function evaluated at NaN will produce an output of NaN if the function depends upon the variable. For example, suppose that $\mathbf{f}(\mathbf{x})$ is a function where $\mathbf{f}:\mathbb{R}^n \longrightarrow \mathbb{R}^m$ and $\mathbf{x} = [\, x_1 \ldots x_n \,]$. Suppose further that we set $x_i = \text{NaN}$. If any of the components of $\mathbf{f}(\mathbf{x})$ is a function of x_i, then those components of $\mathbf{f}(\mathbf{x})$ that depend on x_i will each be NaN. In this way, the dependence of the optimal control problem functions on a particular input component can be determined by evaluating the functions with that component of the input set to NaN using a single function evaluation. Furthermore, the complete first-derivative dependencies of the optimal control problem functions are found by repeating this process for each component of the input. These dependencies then provide a map as to which finite differences need to be taken when computing the first derivatives of the optimal control functions for determining the NLP derivative functions as described in Section 4.4.

Suppose now that we denote $\mathbf{f}(\mathbf{z})$ generically as one of functions defined in the continuous-time optimal control problem of Eqs. (1)–(8) and that $\mathbf{Jf} = \partial \mathbf{f}/\partial \mathbf{z}$ is the

Jacobian of $\mathbf{f}$ with respect to $\mathbf{z}$. Furthermore, suppose that $\mathbf{S}_1 = \mathbf{struct}(\mathbf{Jf})$ is a matrix of ones and zeros that contains the structure the first derivatives of the optimal control function $\mathbf{f}(\mathbf{z})$. The second derivative dependencies are then obtained as the matrix of ones and zeros given by $\mathbf{S}_2 = \mathbf{struct}(\mathbf{S}_1^{\top}\mathbf{S}_1)$. Now, while it may be the case for some problems that $\mathbf{S}_2$ contains an over-estimate of the second derivative dependencies of the function $\mathbf{f}(\mathbf{z})$, for most problems to which $\mathbb{GPOPS}-\mathbb{II}$ may be applied (namely, nonlinear optimal control problems), the estimate obtained in $\mathbf{S}_2$ will be quite close to the exact second derivative dependencies of the function $\mathbf{f}(\mathbf{z})$. In addition, our approach is robust in that it will never under-estimate the second derivative dependencies (where an under-estimate may prove to be catastrophic because second derivatives that are nonzero in the optimal control problem would be incorrectly estimated to be zero in the approximation of the Lagrangian Hessian). Finally, our approach for determining second derivative dependencies incurs minimal additional computational expense above that required to determine the first derivative dependencies.

4.6. Adaptive Mesh Refinement

In the past few years, the subject of variable-order mesh refinement has been a topic of considerable study in developing efficient implementations of Gaussian quadrature collocation methods. The work on variable-order Gaussian quadrature mesh refinement has led to several articles in the literature including those found in Gong et al. [2008a; Darby et al. [2011b; Darby et al. [2011c; Patterson et al. [2013]. $\mathbb{GPOPS}-\mathbb{II}$ employs the two latest variable-order mesh refinement methods as described in Darby et al. [2011c; Patterson et al. [2013]. The mesh refinement methods of Darby et al. [2011c] and Patterson et al. [2013] are referred to as the hp and the ph methods, respectively. In either the hp and the ph mesh refinement methods the number of mesh intervals, width of each mesh interval, and the degree of the approximating polynomial can be varied until a user-specified accuracy tolerance has been achieved. When using either of the methods in $\mathbb{GPOPS}-\mathbb{II}$, the terminology $hp-(N_{\min}, N_{\max})$ or $ph-(N_{\min}, N_{\max})$ refers to a method whose minimum and maximum allowable polynomial degrees within a mesh interval are $N_{\min}$ and $N_{\max}$, respectively. Each method estimates the solution error using a relative difference between the state estimate and the integral of the dynamics at a modified set of LGR points. The key difference between the hp and the ph methods lies in the manner in which the decision is made to either increase the number of collocation points in a mesh interval or to refine the mesh. In Darby et al. [2011c] the degree of the approximating polynomial is increased if the ratio of the maximum curvature over the mean curvature of the state in a particular mesh interval exceeds a user-specified threshold. On the other hand, Patterson et al. [2013] uses the exponential convergence property of the Radau collocation method and increases the polynomial degree within a mesh interval if the estimate of the required polynomial degree is less than a user-specified upper limit. If the estimate of the polynomial degree exceeds the allowed upper limit, the mesh interval is divided into more mesh intervals. In $\mathbb{GPOPS}-\mathbb{II}$ the user can choose between these two mesh refinement methods. Finally, it is noted that $\mathbb{GPOPS}-\mathbb{II}$ has been designed in a modular way making it possible to add a new mesh refinement method in a relatively straightforward way if it is so desired.

4.7. Algorithmic Flow of $\mathbb{GPOPS}-\mathbb{II}$

In this section we describe the operational flow of $\mathbb{GPOPS}-\mathbb{II}$ with the aid of Fig. 7. First, the user provides a description of the optimal control problem that is to be solved. The properties of the optimal control problem are then determined from the user description from which the state, control, time, and parameter dependencies of the optimal control problem functions are determined. Subsequently, assuming that the user

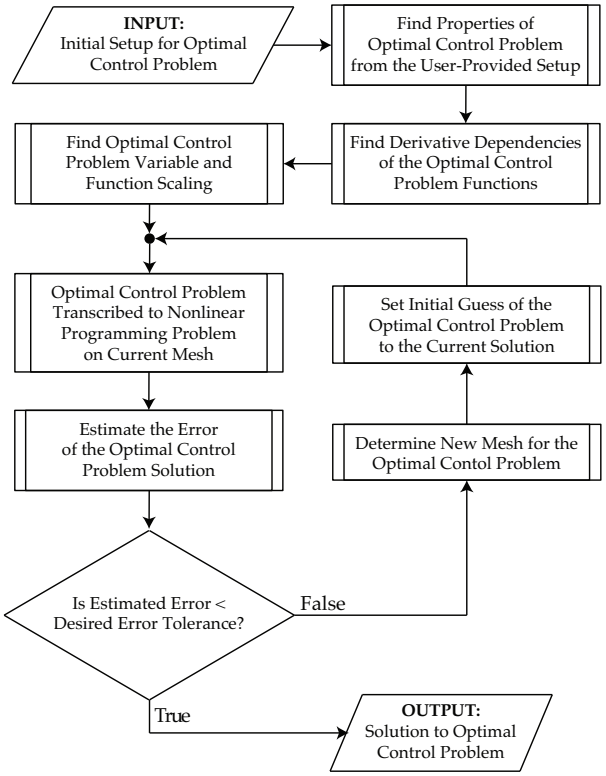


Fig. 7: Flowchart of the $\mathbb{GPOPS} - \mathbb{II}$ Algorithm.

has specified that the optimal control problem be scaled automatically, the optimal control problem scaling algorithm is called and these scale factors are used to scale the NLP. The optimal control problem is then transcribed to a large sparse NLP and the NLP is solved on the initial mesh, where the initial mesh is either user-supplied or is determined by the defaults set in $\mathbb{GPOPS} - \mathbb{II}$. Once the NLP is solved, it is untranscribed to a discrete approximation of the optimal control problem and the error in the discrete approximation for the current mesh is estimated. If the user-specified accuracy tolerance is met, the software terminates and outputs the solution. Otherwise, a new mesh is determined using one of the supplied mesh refinement algorithms and the resulting NLP is solved on the new mesh.

5. EXAMPLES

$\mathbb{GPOPS} - \mathbb{II}$ is now demonstrated on five examples taken from the open literature. The first example is the hyper-sensitive optimal control problem from Ref. Rao and Mease [2000] and demonstrates the ability of $\mathbb{GPOPS} - \mathbb{II}$ to efficiently solve problems that have rapid changes in dynamics in particular regions of the solution. The second example is the reusable launch vehicle entry problem taken from Ref. Betts [2010] and demonstrates the efficiency of $\mathbb{GPOPS} - \mathbb{II}$ on a problem whose dynamic model is representative of a real physical system. The third example is the space station attitude optimal control problem taken from Pietz [2003] and Betts [2010] and demonstrates the efficiency of $\mathbb{GPOPS} - \mathbb{II}$ on a problem whose solution is highly non-intuitive. The fourth example is a kinetic batch reactor problem taken from Betts [2010] and demon-

strates the ability of $\mathbb{GPOPS}-\mathbb{II}$ to solve an extremely badly scaled multiple-phase optimal control problem. The fifth example is a multiple-stage launch vehicle ascent problem taken from Benson [2004; Rao et al. [2010; Betts [2010] and demonstrates the ability of $\mathbb{GPOPS}-\mathbb{II}$ to solve a problem with multiple-phases. The first four examples were solved using the open-source NLP solver IPOPT [Biegler et al. 2003] in second derivative (full Newton) mode with the publicly available multifrontal massively parallel sparse direct linear solver MUMPS [MUMPS 2011], while the fifth example was solved using the NLP solver SNOPT [Gill et al. 2002]. The initial guess of the solution is provided for all examples, where the function $\mathbf{linear}(t_0, t_f, a, b)$ is a linear interpolation over the range $[a, b]$ on the domain $[t_0, t_f]$. All results were obtained using the implicit integration form of the Radau collocation method and various forms of the aforementioned ph mesh refinement method using default NLP solver settings and the automatic scaling routine in $\mathbb{GPOPS}-\mathbb{II}$.

5.1. Hyper-Sensitive Problem

Consider the following *hyper-sensitive* optimal control problem taken from Rao and Mease [2000]:

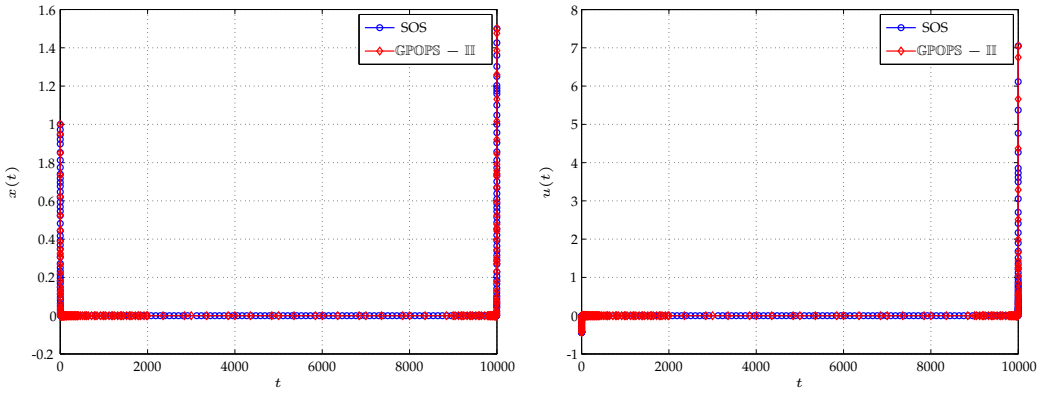
$$\text{minimize } \tfrac{1}{2}\int_0^{t_f}(x^2+u^2)dt \text{ subject to } \begin{cases} \dot{x} &= -x^3+u, \\ x(0) &= 1, \\ x(t_f) &= 1.5, \end{cases} \tag{59}$$

where $t_f = 10000$. It is known for a sufficiently large value of t_f the interesting behavior in the solution occurs near $t = 0$ and $t = t_f$ (see Ref. [Rao and Mease 2000] for details), while the vast majority of the solution is a constant. Given the structure of the solution, a majority of collocation points need to be placed near $t = 0$ and $t = t_f$.

The hyper-sensitive optimal control problem was solved using $\mathbb{GPOPS}-\mathbb{II}$ with the $ph-(3,10)$ method, an initial mesh consisting of ten evenly spaced mesh interval with three LGR points per mesh interval, and the following initial guess:

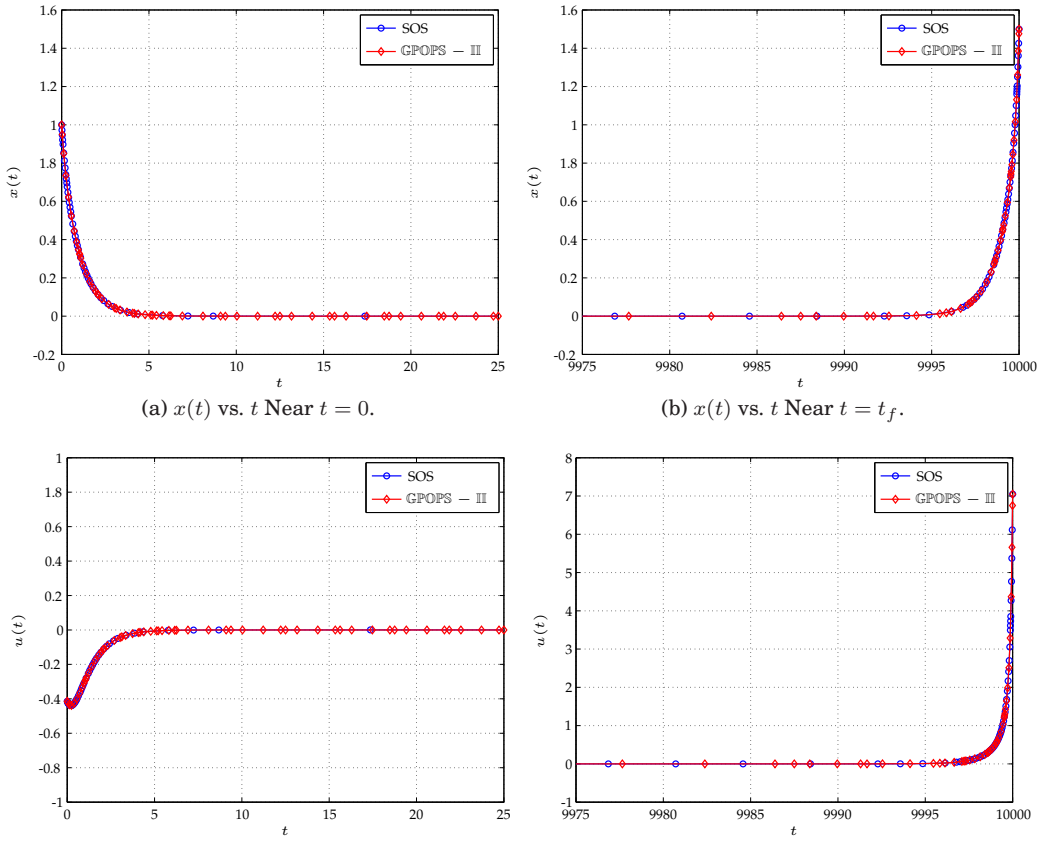
$$\begin{aligned} x^{\text{guess}} &= \mathbf{linear}(0, t_f, x(0), x(t_f)), \\ u^{\text{guess}} &= 0, \\ t_f^{\text{guess}} &= t_f. \end{aligned}$$

The solution obtained using $\mathbb{GPOPS}-\mathbb{II}$ is shown in Fig. 8 alongside the solution obtained with the software *Sparse Optimization Suite* (SOS) [Betts 2013]. It is seen that the $\mathbb{GPOPS}-\mathbb{II}$ and SOS solutions are in excellent agreement. Moreover, the optimal cost obtained using $\mathbb{GPOPS}-\mathbb{II}$ and SOS are extremely close, with values 3.3620563 and 3.3620608, respectively. In order to demonstrate how $\mathbb{GPOPS}-\mathbb{II}$ is capable of capturing the interesting features of the optimal solution, Fig. 9 shows the solution on the intervals $t \in [0, 15]$ (near the initial time) and $t \in [9985, 10000]$ (near the final time), while Fig. 10 shows the mesh refinement history. It is seen that $\mathbb{GPOPS}-\mathbb{II}$ accurately captures the rapid decay from $x(0) = 1$ and the rapid growth to meet the terminal condition $x(t_f) = 1.5$, and the density of the mesh points near $t = 0$ and $t = t_f$ increases as the mesh refinement progresses. Finally, Table I shows the estimated error on each mesh, where it is seen that the solution error decreases steadily with each mesh refinement iteration, finally terminating on the eighth mesh (that is, the seventh mesh refinement).



(a) $x(t)$ vs. t. (b) $u(t)$ vs. t.

Fig. 8: $\mathbb{GPOPS} - \mathbb{II}$ and *Sparse Optimization Suite* Solutions to Hyper-Sensitive Optimal Control Problem.



(a) $x(t)$ vs. t Near $t = 0$. (b) $x(t)$ vs. t Near $t = t_f$.

(c) $u(t)$ vs. t Near $t = 0$. (d) $u(t)$ vs. t Near $t = t_f$.

Fig. 9: $\mathbb{GPOPS} - \mathbb{II}$ and *Sparse Optimization Suite* Solutions to Hyper-Sensitive Optimal Control Problem Near $t = 0$ and $t = t_f$.

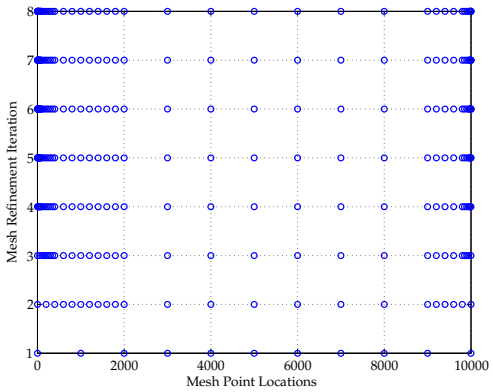


Fig. 10: Mesh Refinement History for Hyper-Sensitive Problem Using $\mathbb{GPOPS}-\mathbb{II}$.

Table I: Mesh Refinement History for Hyper-Sensitive Problem.

Mesh	**Relative Error Estimate**
1	2.827×10^{1}
2	2.823×10^{0}
3	7.169×10^{-1}
4	1.799×10^{-1}
5	7.092×10^{-2}
6	8.481×10^{-3}
7	1.296×10^{-3}
8	5.676×10^{-7}

5.2. Reusable Launch Vehicle Entry

Consider the following optimal control problem of maximizing the crossrange during the atmospheric entry of a reusable launch vehicle and taken from Betts [2010] where the numerical values in Betts [2010] are converted from English units to SI units. Maximize the cost functional

$$J = \phi(t_f) \tag{60}$$

subject to the dynamic constraints

$$\begin{aligned}
&\dot{r} = v\sin\gamma \quad , \quad \dot{\theta} = \frac{v\cos\gamma\sin\psi}{r\cos\phi} \quad , \quad \dot{\phi} = \frac{v\cos\gamma\cos\psi}{r}, \\
&\dot{v} = -\frac{D}{m} - g\sin\gamma \quad , \quad \dot{\gamma} = \frac{L\cos\sigma}{mv} - \left(\frac{g}{v} - \frac{v}{r}\right)\cos\gamma \quad , \quad \dot{\psi} = \frac{L\sin\sigma}{mv\cos\gamma} + \frac{v\cos\gamma\sin\psi\tan\phi}{r},
\end{aligned} \tag{61}$$

and the boundary conditions

$$\begin{array}{llllllll}
h(0) = 79248\ \text{km} & , & h(t_f) = 24384\ \text{km} & , & \theta(0) = 0\ \text{deg} & , & \theta(t_f) = \text{Free}, \\
\phi(0) = 0\ \text{deg} & , & \phi(t_f) = \text{Free} & , & v(0) = 7.803\ \text{km/s} & , & v(t_f) = 0.762\ \text{km/s} \\
\gamma(0) = -1\ \text{deg} & , & \gamma(t_f) = -5\ \text{deg} & , & \psi(0) = 90\ \text{deg} & , & \psi(t_f) = \text{Free},
\end{array} \tag{62}$$

where $r = h + R_e$ is the geocentric radius, h is the altitude, R_e is the polar radius of the Earth, θ is the longitude, ϕ is the latitude, v is the speed, γ is the flight path angle, and ψ is the azimuth angle. Furthermore, the aerodynamic and gravitational forces are computed as

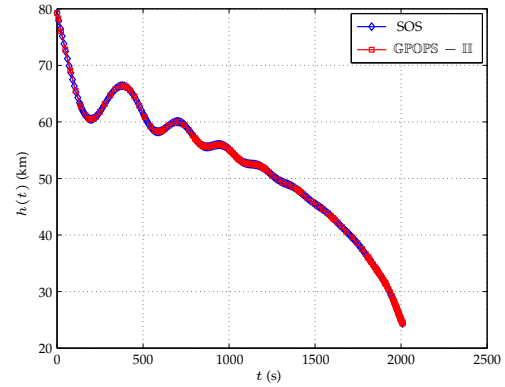
$$D = \rho v^2 S C_D/2 \ , \ L = \rho v^2 S C_L/2 \ , \ g = \mu/r^2, \tag{63}$$

where $\rho = \rho_0 \exp(-h/H)$ is the atmospheric density, ρ_0 is the density at sea level, H is the density scale height, S is the vehicle reference area, C_D is the coefficient of drag, C_L is the coefficient of lift, and μ is the gravitational parameter.

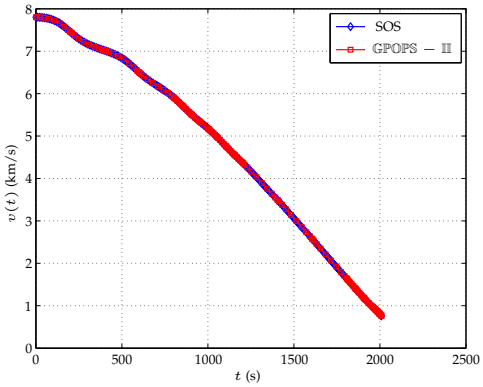
The reusable launch vehicle entry optimal control problem was solved with $\mathbb{GPOPS}-\mathbb{II}$ using the $ph-(4,10)$ mesh refinement method, an initial mesh consisting of ten evenly spaced mesh intervals with four LGR points per mesh interval, and a mesh refinement accuracy tolerance of 10^{-7}, and the following initial guess:

$$\begin{aligned}
h &= \textbf{linear}(0, t_f, h(0), h(t_f)), \\
\theta &= \theta(0), \\
\phi &= \phi(0), \\
v &= \textbf{linear}(0, t_f, v(0), v(t_f)), \\
\gamma &= \textbf{linear}(0, t_f, \gamma(0), \gamma(t_f)), \\
\psi &= \psi(0), \\
\alpha &= 0, \\
\sigma &= 0, \\
t_f &= 1000\ \text{s}.
\end{aligned}$$

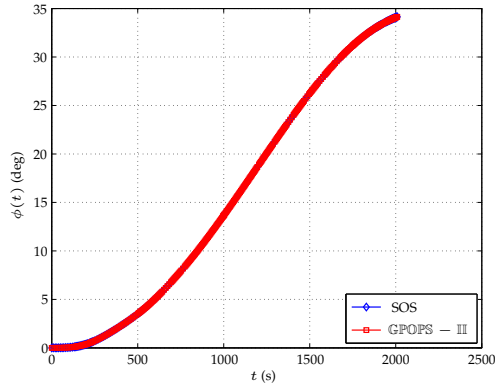
The solution obtained using $\mathbb{GPOPS}-\mathbb{II}$ is shown in Figs. 11a–11f alongside the solution obtained using the software *Sparse Optimization Suite* (SOS) [Betts 2010], where it is seen that the two solutions obtained are virtually indistinguishable. It is noted that the optimal cost obtained by $\mathbb{GPOPS}-\mathbb{II}$ and SOS are also nearly identical at 0.59627639 and 0.59587608, respectively. Table II shows the performance of both $\mathbb{GPOPS}-\mathbb{II}$ and SOS on this example. It is interesting to see that $\mathbb{GPOPS}-\mathbb{II}$ meets the accuracy tolerance of 10^{-7} in only four mesh iterations (three mesh refinements) while SOS requires a total of eight meshes (seven mesh refinements). Finally, the number of collocation points used by $\mathbb{GPOPS}-\mathbb{II}$ is approximately one half the number of collocation points required by SOS to achieve the same level of accuracy.



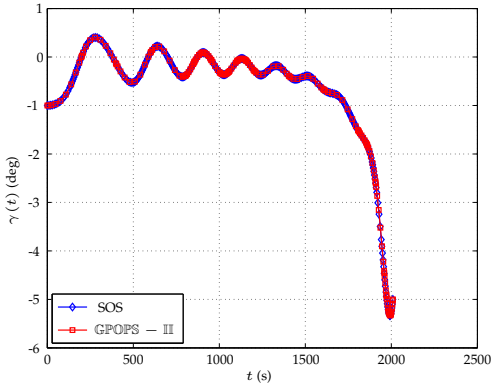
(a) $h(t)$ vs. t.



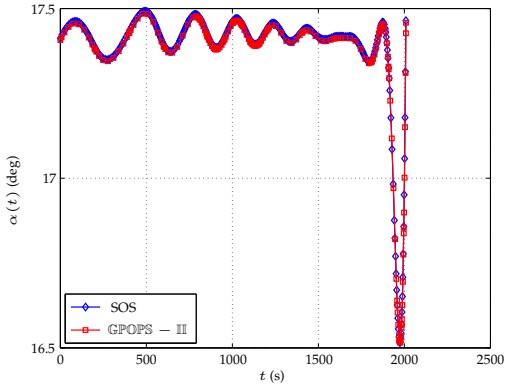
(b) $v(t)$ vs. t.



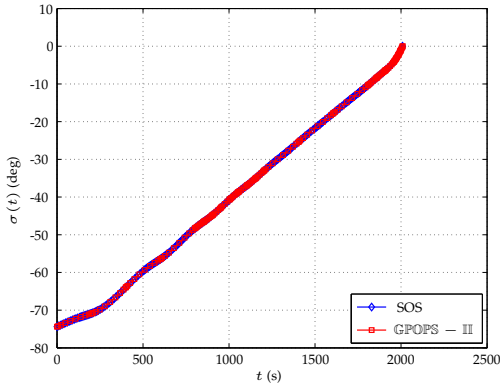
(c) $\phi(t)$ vs. t.



(d) $\gamma(t)$ vs. t.



(e) $\alpha(t)$ vs. t.



(f) $\sigma(t)$ vs. t.

Fig. 11: Solution to Reusable Launch Vehicle Entry Problem Using GPOPS − II.

Table II: Performance of $\mathbb{GPOPS}-\mathbb{II}$ on the Reusable Launch Vehicle Entry Optimal Control Problem.

Mesh Iteration	Estimated Error ($\mathbb{GPOPS}-\mathbb{II}$)	Number of Collocation Points	Estimated Error (SOS)	Number of Collocation Points
1	2.463×10^{-3}	41	1.137×10^{-2}	51
2	2.946×10^{-4}	103	1.326×10^{-3}	101
3	1.202×10^{-5}	132	3.382×10^{-5}	101
4	8.704×10^{-8}	175	1.314×10^{-6}	101
5	–	–	2.364×10^{-7}	201
6	–	–	2.364×10^{-7}	232
7	–	–	1.006×10^{-7}	348
8	–	–	9.933×10^{-8}	353

5.3. Space Station Attitude Control

Consider the following space station attitude control optimal control problem taken from Pietz [2003] and Betts [2010]. Minimize the cost functional

$$J=\tfrac{1}{2}\int_{t_0}^{t_f}\mathbf{u}^{\mathsf{T}}\mathbf{u}dt \tag{64}$$

subject to the dynamic constraints

$$\begin{aligned}\dot{\boldsymbol{\omega}} &= \mathbf{J}^{-1}\left\{\boldsymbol{\tau}_{gg}(\mathbf{r})-\boldsymbol{\omega}^{\otimes}\left[\mathbf{J}\boldsymbol{\omega}+\mathbf{h}\right]-\mathbf{u}\right\},\\ \dot{\mathbf{r}} &= \tfrac{1}{2}\left[\mathbf{r}\mathbf{r}^{\mathsf{T}}+\mathbf{I}+\mathbf{r}\right]\left[\boldsymbol{\omega}-\boldsymbol{\omega}(\mathbf{r})\right],\\ \dot{\mathbf{h}} &= \mathbf{u},\end{aligned} \tag{65}$$

the inequality path constraint

$$\|\mathbf{h}\|\leq h_{\max}, \tag{66}$$

and the boundary conditions

$$\begin{aligned}t_0 &= 0,\\ t_f &= 1800,\\ \boldsymbol{\omega}(0) &= \bar{\boldsymbol{\omega}}_0,\\ \mathbf{r}(0) &= \bar{\mathbf{r}}_0,\\ \mathbf{h}(0) &= \bar{\mathbf{h}}_0,\\ \mathbf{0} &= \mathbf{J}^{-1}\left\{\boldsymbol{\tau}_{gg}(\mathbf{r}(t_f))-\boldsymbol{\omega}^{\otimes}(t_f)\left[\mathbf{J}\boldsymbol{\omega}(t_f)+\mathbf{h}(t_f)\right]\right\},\\ \mathbf{0} &= \tfrac{1}{2}\left[\mathbf{r}(t_f)\mathbf{r}^{\mathsf{T}}(t_f)+\mathbf{I}+\mathbf{r}(t_f)\right]\left[\boldsymbol{\omega}(t_f)-\boldsymbol{\omega}_0(\mathbf{r}(t_f))\right],\end{aligned} \tag{67}$$

where $(\boldsymbol{\omega},\mathbf{r},\mathbf{h})$ is the state and $\mathbf{u}$ is the control. In this formulation $\boldsymbol{\omega}$ is the angular velocity, $\mathbf{r}$ is the Euler-Rodrigues parameter vector, $\mathbf{h}$ is the angular momentum, and $\mathbf{u}$ is the input moment (and is the control).

$$\begin{aligned}\boldsymbol{\omega}_0(\mathbf{r}) &= -\omega_{\text{orb}}\mathbf{C}_2,\\ \boldsymbol{\tau}_{gg} &= 3\omega_{\text{orb}}^2\mathbf{C}_3^{\otimes}\mathbf{J}\mathbf{C}_3,\end{aligned} \tag{68}$$

and $\mathbf{C}_2$ and $\mathbf{C}_3$ are the second and third column, respectively, of the matrix

$$\mathbf{C}=\mathbf{I}+\frac{2}{1+\mathbf{r}^{\mathsf{T}}\mathbf{r}}\left(\mathbf{r}^{\otimes}\mathbf{r}^{\otimes}-\mathbf{r}^{\otimes}\right). \tag{69}$$

In this example the matrix $\mathbf{J}$ is given as

$$\mathbf{J}=\begin{bmatrix}2.80701911616\times10^{7} & 4.822509936\times10^{5} & -1.71675094448\times10^{7}\\ 4.822509936\times10^{5} & 9.5144639344\times10^{7} & 6.02604448\times10^{4}\\ -1.71675094448\times10^{7} & 6.02604448\times10^{4} & 7.6594401336\times10^{7}\end{bmatrix}, \tag{70}$$

while the initial conditions $\bar{\boldsymbol{\omega}}_0$, $\bar{\mathbf{r}}_0$, and $\bar{\mathbf{h}}_0$ are

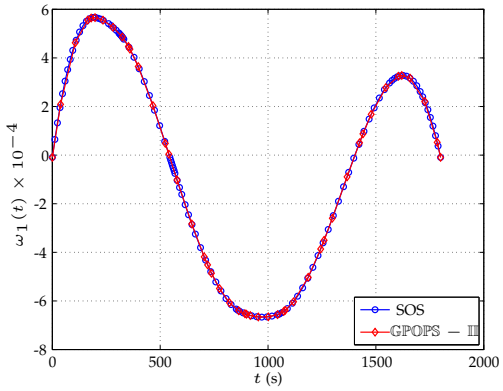
$$\begin{aligned}
\bar{\boldsymbol{\omega}}_0 &= \begin{bmatrix} -9.5380685844896 \times 10^{-6} \\ -1.1363312657036 \times 10^{-3} \\ +5.3472801108427 \times 10^{-6} \end{bmatrix}, \\
\bar{\mathbf{r}}_0 &= \begin{bmatrix} 2.9963689649816 \times 10^{-3} \\ 1.5334477761054 \times 10^{-1} \\ 3.8359805613992 \times 10^{-3} \end{bmatrix}, \\
\bar{\mathbf{h}}_0 &= \begin{bmatrix} 5000 \\ 5000 \\ 5000 \end{bmatrix}.
\end{aligned} \tag{71}$$

A more detailed description of this problem, including all of the constants $\mathbf{J}$, $\bar{\boldsymbol{\omega}}_0$, $\bar{\mathbf{r}}_0$, and $\bar{\mathbf{h}}_0$, can be found in Pietz [2003] or Betts [2010].

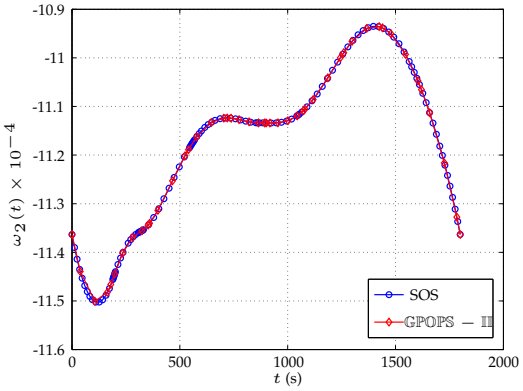
The space station attitude control example was solved with $\mathbb{GPOPS}-\mathbb{II}$ using the $ph-(4,10)$ mesh refinement method with an initial mesh consisting of ten uniformly spaced mesh intervals and four LGR points per mesh interval, a finite-difference perturbation step size of 10^{-5}, and the following initial guess:

$$\begin{aligned}
\boldsymbol{\omega} &= \bar{\boldsymbol{\omega}}_0 \\
\mathbf{r} &= \bar{\mathbf{r}}_0 \\
\mathbf{h} &= \bar{\mathbf{h}}_0 \\
\mathbf{u} &= \mathbf{0}, \\
t_f &= 1800 \text{ s}.
\end{aligned}$$

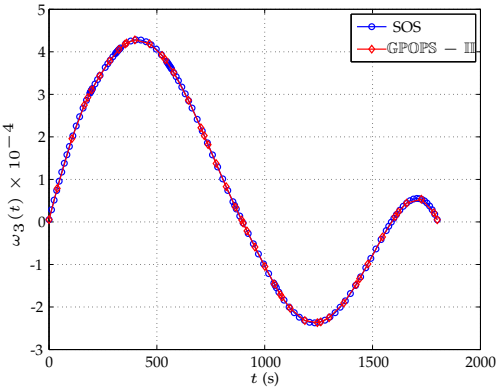
The state and control solutions obtained using $\mathbb{GPOPS}-\mathbb{II}$ are shown, respectively, in Fig. 12 and 13 alongside the solution obtained using the optimal control software *Sparse Optimization Suite* (SOS) [Betts 2013]. It is seen that the $\mathbb{GPOPS}-\mathbb{II}$ solution is in close agreement with the SOS solution. It is noted for this example that the mesh refinement accuracy tolerance of 10^{-6} was satisfied on the second mesh (that is, one mesh refinement iteration was performed) using a total of 46 collocation (LGR) points (that is, 47 total points when including the final time point).



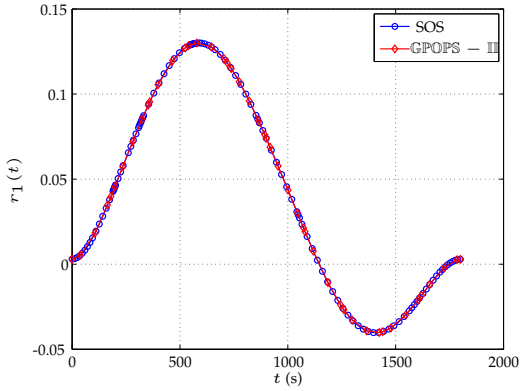
(a) $\omega_1(t)$ vs. t.



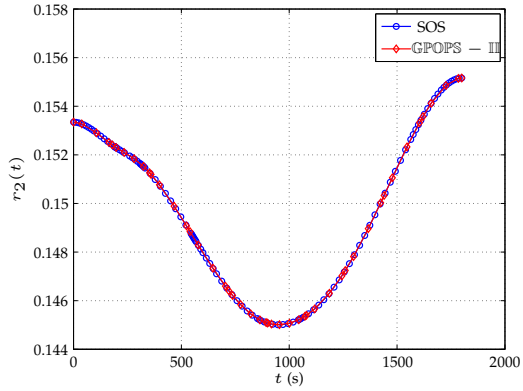
(b) $\omega_2(t)$ vs. t.



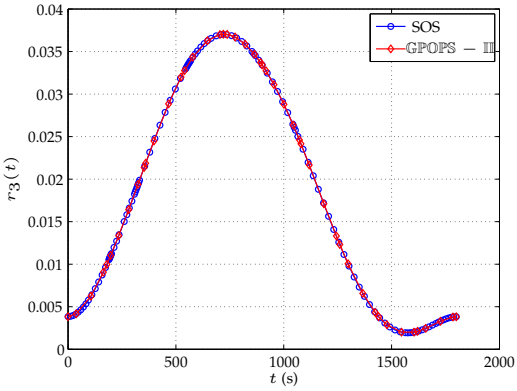
(c) $\omega_3(t)$ vs. t.



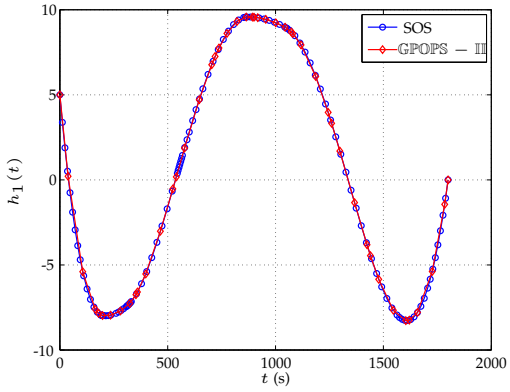
(d) $r_1(t)$ vs. t.



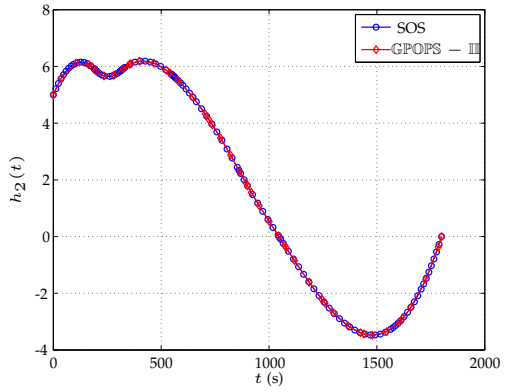
(e) $r_2(t)$ vs. t.



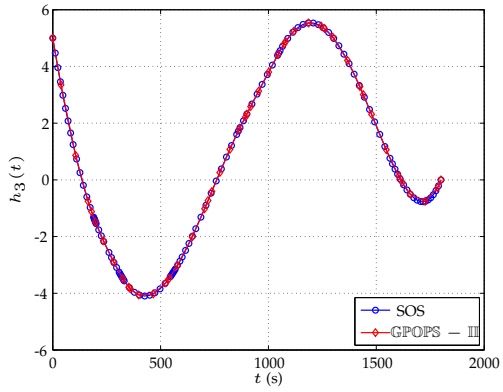
(f) $r_3(t)$ vs. t.



(g) $h_1(t)$ vs. t.

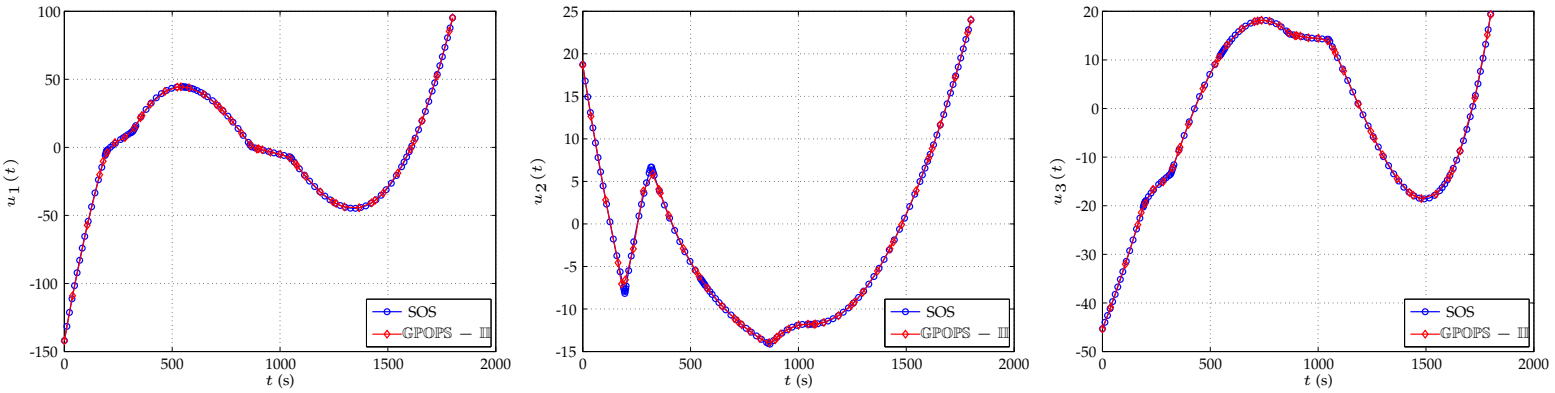


(h) $h_2(t)$ vs. t.



(i) $h_3(t)$ vs. t.

Fig. 12: State Solution to Space Station Attitude Control Problem Using $\mathbb{GPOPS} - \mathbb{II}$ with the NLP Solver IPOPT and a Mesh Refinement Tolerance of 10^{-6} Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.



(a) $u_1(t)$ vs. t. (b) $u_2(t)$ vs. t. (c) $u_3(t)$ vs. t.

Fig. 13: Control Solution to Space Station Attitude Control Problem Using $\mathbb{GPOPS} - \mathbb{II}$ with the NLP Solver IPOPT and a Mesh Refinement Tolerance of 10^{-6} Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.

5.4. Kinetic Batch Reactor

Consider the following three-phase kinetic batch reactor optimal control problem that originally appears in the work of Leineweber [1998] and later appears in Betts [2010]. Minimize the cost functional

$$J = \gamma_1 t_f^{(3)} + \gamma_2 p \tag{72}$$

subject to the dynamic constraints

(73)

$$\begin{aligned}
\dot{y}_1^{(k)} &= -k_2 y_2^{(k)} u_2^{(k)}, \\
\dot{y}_2^{(k)} &= -k_1 y_2^{(k)} y_6^{(k)} + k_{-1} u_4^{(k)} - k_2 y_2^{(k)} u_4^{(k)}, \\
\dot{y}_3^{(k)} &= k_2 y_2^{(k)} u_2^{(k)} + k_3 y_4^{(k)} y_6^{(k)} - k_{-3} u_3^{(k)}, \\
\dot{y}_4^{(k)} &= -k_3 y_4^{(k)} y_6^{(k)} + k_{-3} u_3^{(k)}, \\
\dot{y}_5^{(k)} &= k_1 y_2^{(k)} y_6^{(k)} - k_{-1} u_4^{(k)}, \\
\dot{y}_6^{(k)} &= -k_1 y_2^{(k)} y_6^{(k)} + k_{-1} u_4^{(k)} - k_3 y_4^{(k)} y_6^{(k)} + k_{-3} u_3^{(k)},
\end{aligned} \quad , \quad (k = 1, 2, 3), \tag{74}$$

the equality path constraints

$$\begin{aligned}
p - y_6^{(k)} + 10^{-u_1^{(k)}} - u_2^{(k)} - u_3^{(k)} - u_4^{(k)} &= 0, \\
u_2^{(k)} - K_2 y_1^{(k)} / (K_2 + 10^{-u_1^{(k)}}) &= 0, \\
u_3^{(k)} - K_3 y_3^{(k)} / (K_3 + 10^{-u_1^{(k)}}) &= 0, \\
u_4^{(k)} - K_4 y_5 / (K_1 + 10^{-u_1^{(k)}}) &= 0,
\end{aligned} , (k = 1, 2, 3), \tag{75}$$

the control inequality path constraint

$$293.15 \leq u_5^{(k)} \leq 393.15, \quad (k = 1, 2, 3), \tag{76}$$

the inequality path constraint in phases 1 and 2

$$y_4^{(k)} \leq a\left[t^{(k)}\right]^2, \quad (k = 1, 2), \tag{77}$$

the interior point constraints

$$\begin{array}{rcl} t_f^{(1)} & = & 0.01, \\ t_f^{(2)} & = & t_f^{(3)}/4, \\ y_i^{(k)} & = & y_i^{(k+1)}, \quad (i = 1, \ldots, 6,\ k = 1, 2, 3), \end{array} \tag{78}$$

and the boundary conditions

$$\begin{array}{lll} y_1^{(1)}(0) = 1.5776, & y_2^{(1)}(0) = 8.32, & y_3^{(1)}(0) = 0, \\ y_4^{(1)}(0) = 0, & y_5^{(1)}(0) = 0, & y_6^{(1)}(0) - p = 0, \\ & y_4^{(3)}(t_f^{(3)}) \leq 1, & \end{array} \tag{79}$$

where

$$\begin{array}{rcl} k_1 & = & \hat{k}_1 \exp(-\beta_1/u_5^{(k)}), \\ k_{-1} & = & \hat{k}_{-1} \exp(-\beta_{-1}/u_5^{(k)}), \\ k_2 & = & \hat{k}_2 \exp(-\beta_2/u_5^{(k)}), \quad , \quad (k = 1, 2, 3), \\ k_3 & = & k_1, \\ k_{-3} & = & \frac{1}{2}k_{-1}, \end{array} \tag{80}$$

and the values for the parameters $\hat{k}_j$, β_j, and K_j are given as

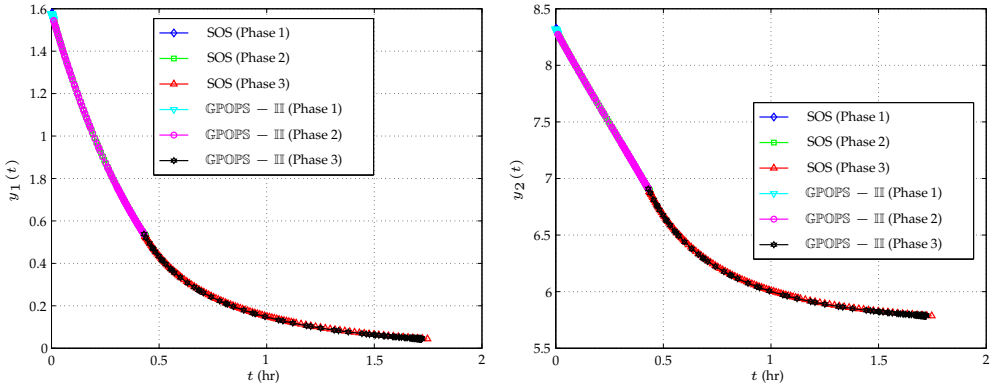
$$\begin{array}{rclrclrcl} \hat{k}_1 & = & 1.3708 \times 10^{12} & , \ \beta_1 & = & 9.2984 \times 10^3 & , \ K_1 & = & 2.575 \times 10^{-16}, \\ \hat{k}_{-1} & = & 1.6215 \times 10^{20} & , \ \beta_{-1} & = & 1.3108 \times 10^4 & , \ K_2 & = & 4.876 \times 10^{-14}, \\ \hat{k}_2 & = & 5.2282 \times 10^{12} & , \ \beta_2 & = & 9.599 \times 10^3 & , \ K_3 & = & 1.7884 \times 10^{-16}. \end{array} \tag{81}$$

The kinetic batch reactor optimal control problem was solved using $\mathbb{GPOPS} - \mathbb{II}$ using the $ph - (3, 6)$ mesh refinement method with an initial mesh in each phase consisting of ten uniformly spaced mesh intervals with three LGR points per mesh interval, a base derivative perturbation step size of 10^{-5}, and the following linear initial guess

(where the superscript represents the phase number):

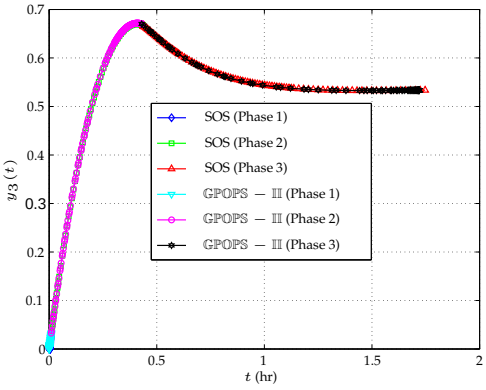
$$
\begin{aligned}
y_1^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, y_1(0), 0.5),\\
y_2^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, y_2(0), 6),\\
y_3^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, y_3(0), 0.6),\\
y_4^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, y_4(0), 0.5),\\
y_5^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, y_5(0), 0.5),\\
y_6^{(1)} &= 0.013,\\
u_1^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, 7, 10),\\
u_2^{(1)} &= 0,\\
u_3^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, 0, 10^{-5}),\\
u_4^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, 0, 10^{-5}),\\
u_5^{(1)} &= \textbf{linear}(t_0^{(1)}, t_f^{(1)}, 373, 393.15).\\
t_0^{(1)} &= 0,\\
t_f^{(1)} &= 0.01 \text{ hr},
\end{aligned}
\qquad
\begin{aligned}
y_1^{(2)} &= 0.5,\\
y_2^{(2)} &= 6,\\
y_3^{(2)} &= 0.6,\\
y_4^{(2)} &= 0.5,\\
y_5^{(2)} &= 0.5,\\
y_6^{(2)} &= 0.013,\\
u_1^{(2)} &= 10,\\
u_2^{(2)} &= 0,\\
u_3^{(2)} &= 10^{-5},\\
u_4^{(2)} &= 10^{-5},\\
u_5^{(2)} &= 393.15,\\
t_0^{(2)} &= 0.01 \text{ hr},\\
t_f^{(2)} &= 2.5 \text{ hr},
\end{aligned}
\qquad
\begin{aligned}
y_1^{(3)} &= \textbf{linear}(t_0^{(3)}, t_f^{(3)}, 0.5, 0.1),\\
y_2^{(3)} &= \textbf{linear}(t_0^{(3)}, t_f^{(3)}, 6, 5),\\
y_3^{(3)} &= 6,\\
y_4^{(3)} &= \textbf{linear}(t_0^{(3)}, t_f^{(3)}, 0.5, 0.9),\\
y_5^{(3)} &= \textbf{linear}(t_0^{(3)}, t_f^{(3)}, 0.5, 0.9),\\
y_6^{(3)} &= 0.013,\\
u_1^{(3)} &= 10,\\
u_2^{(3)} &= 0,\\
u_3^{(3)} &= 10^{-5},\\
u_4^{(3)} &= 10^{-5},\\
u_5^{(3)} &= 393.15,\\
t_0^{(3)} &= 2.5 \text{ hr},\\
t_f^{(3)} &= 10 \text{ hr}.
\end{aligned}
$$

The state and control solutions obtained using $\mathbb{GPOPS}-\mathbb{II}$ are shown in Figs. 14–17, respectively, alongside the solution obtained using the software *Sparse Optimization Suite* (SOS) Betts [2013]. It is seen that in the complete three-phase problem the $\mathbb{GPOPS}-\mathbb{II}$ and SOS solutions have the same trend, the key difference being that the $\mathbb{GPOPS}-\mathbb{II}$ solution is shorter in duration than the SOS solution. A closer examination, however, of the solution in phase 1 reveals that the two solutions are actually quite different at the start of the problem. The $\mathbb{GPOPS}-\mathbb{II}$ solution moves away from the initial condition much more quickly than the SOS solution. Thus, the SOS solution exhibits stiff behavior at the start of the solution while the $\mathbb{GPOPS}-\mathbb{II}$ solution does not exhibit this stiffness. While the solutions are significantly different in phase 1, the optimal cost obtained using $\mathbb{GPOPS}-\mathbb{II}$ is 3.1650187 while the optimal cost obtained using SOS is 3.1646696, leading to absolute and relative differences of only 3.4910×10^{-4} and 1.1031×10^{-4}, respectively. Thus, while the solutions obtained by each software program differ in the first (transient) phase, the overall performance of $\mathbb{GPOPS}-\mathbb{II}$ is similar to that obtained using SOS (particularly given the computational challenge of this example).

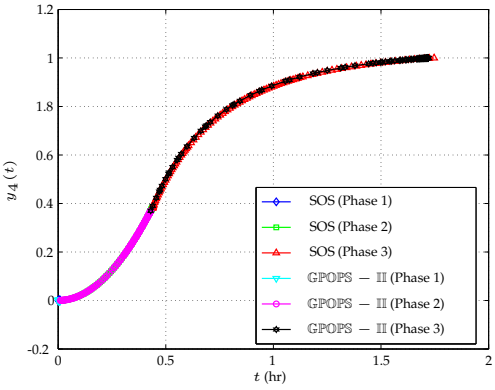


(a) $y_1(t)$ vs. t.

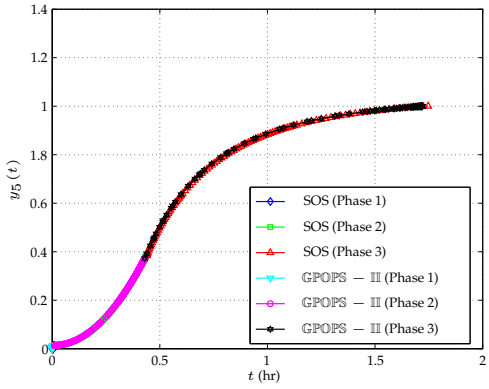
(b) $y_2(t)$ vs. t.



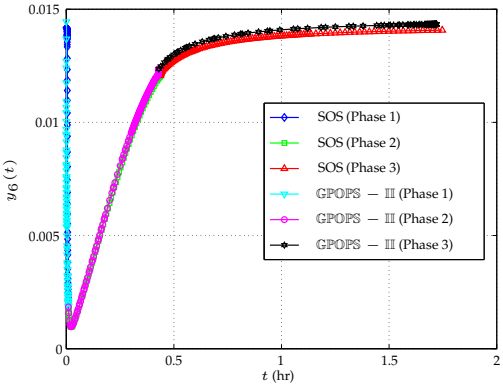
(c) $y_3(t)$ vs. t.



(d) $y_4(t)$ vs. t.

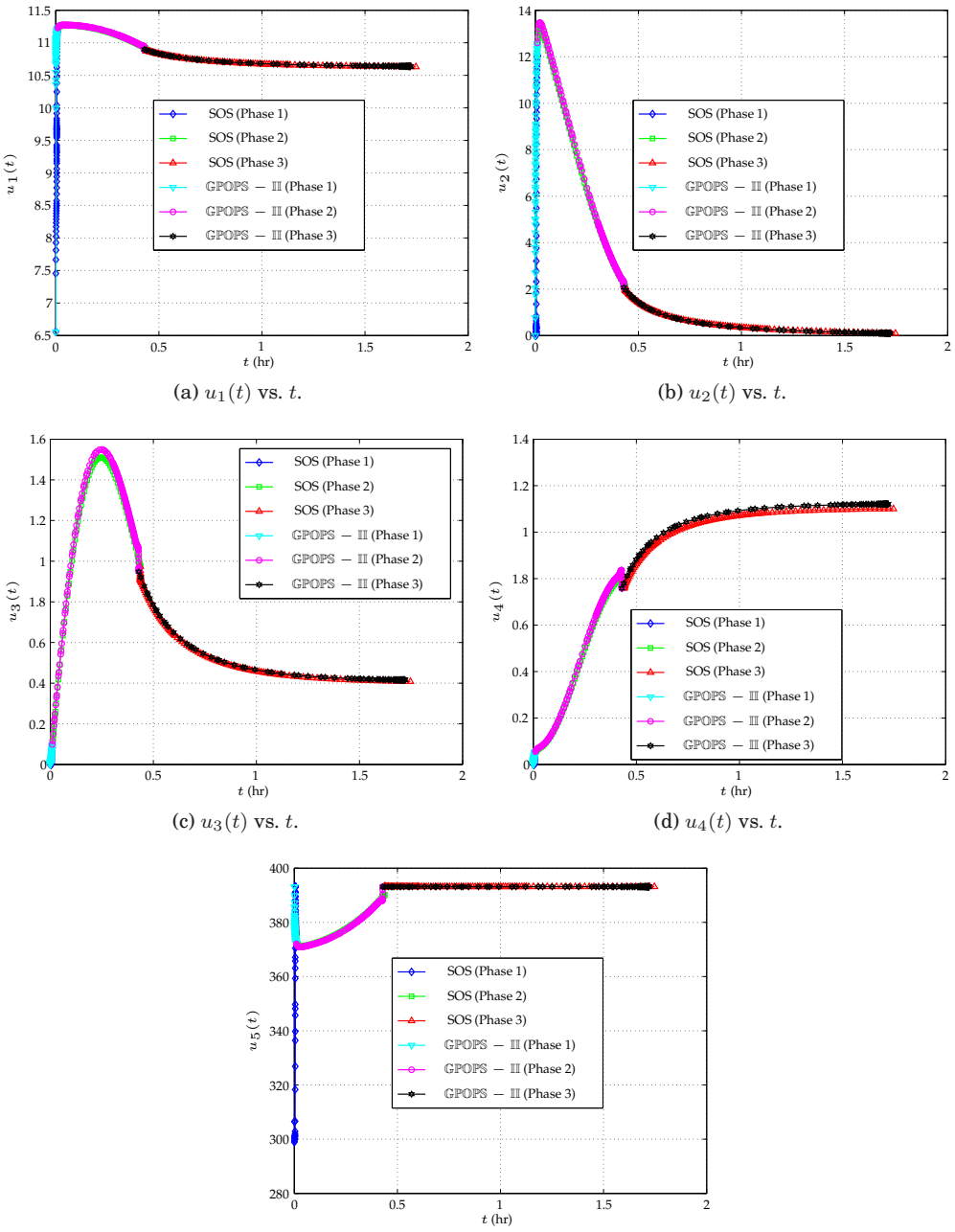


(e) $y_5(t)$ vs. t.



(f) $y_6(t)$ vs. t.

Fig. 14: State Solution to Kinetic Batch Reactor Optimal Control Problem Using GPOPS – II Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.



(a) $u_1(t)$ vs. t.

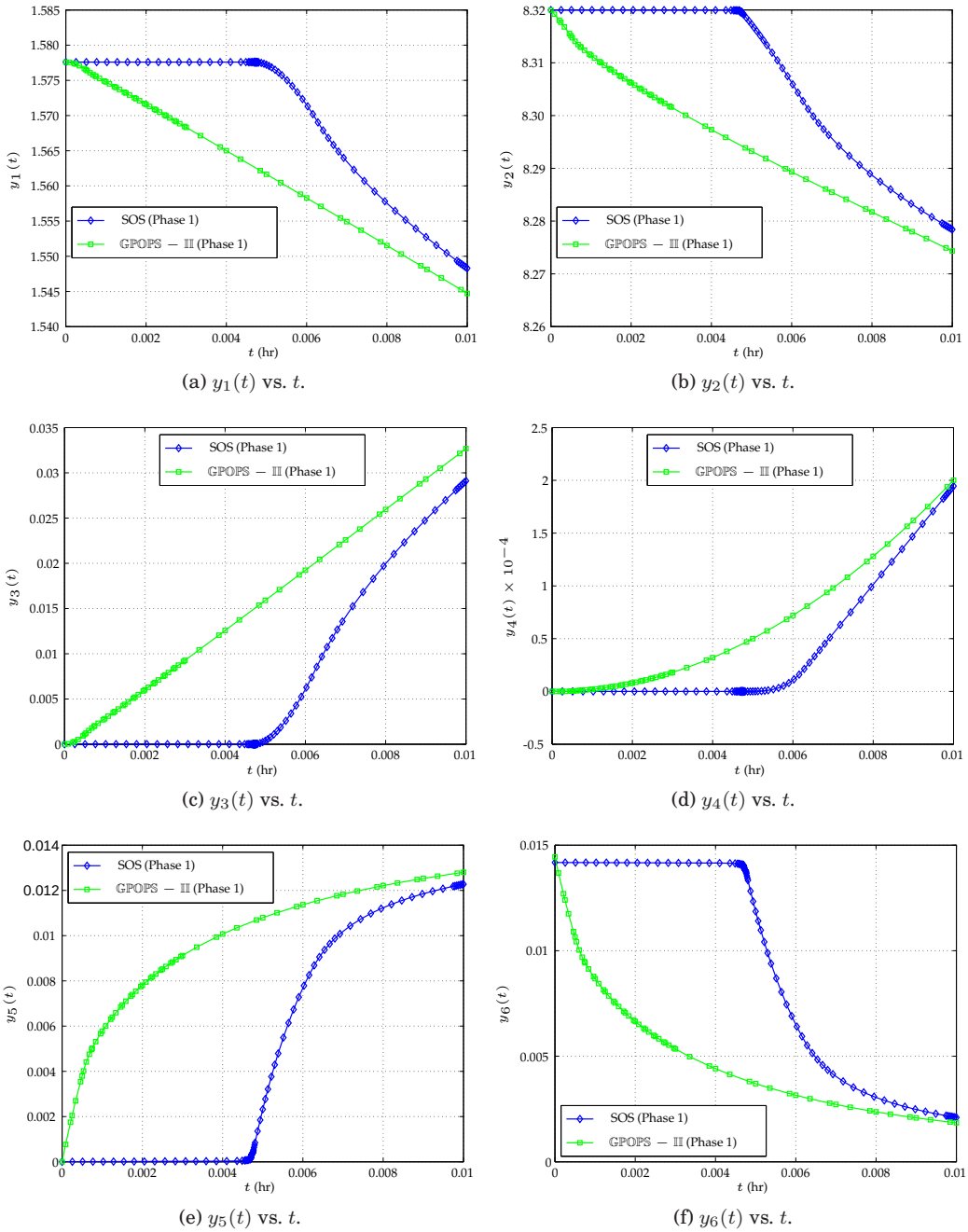
(b) $u_2(t)$ vs. t.

(c) $u_3(t)$ vs. t.

(d) $u_4(t)$ vs. t.

(e) $u_5(t)$ vs. t.

Fig. 15: Control Solution to Kinetic Batch Reactor Optimal Control Problem Using GPOPS – II Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.



(a) $y_1(t)$ vs. t.

(b) $y_2(t)$ vs. t.

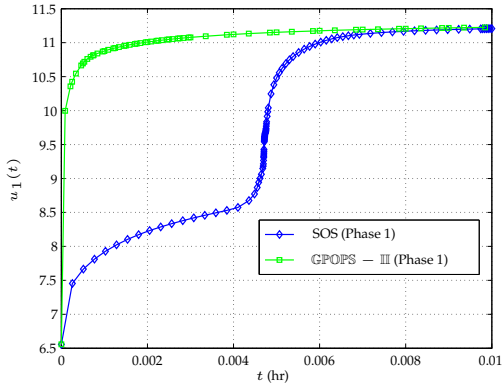
(c) $y_3(t)$ vs. t.

(d) $y_4(t)$ vs. t.

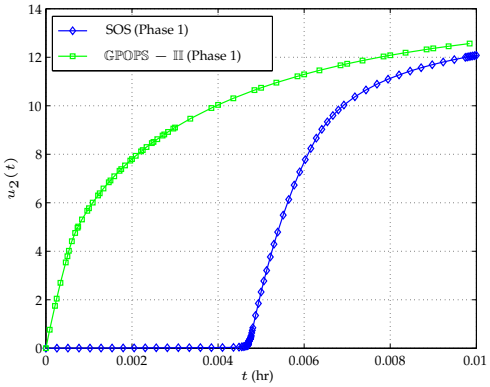
(e) $y_5(t)$ vs. t.

(f) $y_6(t)$ vs. t.

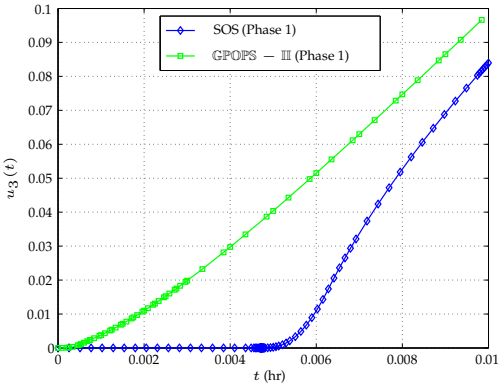
Fig. 16: Phase 1 State Solution to Kinetic Batch Reactor Optimal Control Problem Using $\mathbb{GPOPS} - \mathbb{II}$ Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.



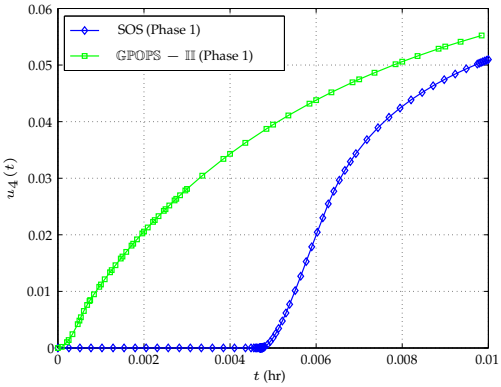
(a) $u_1(t)$ vs. t.



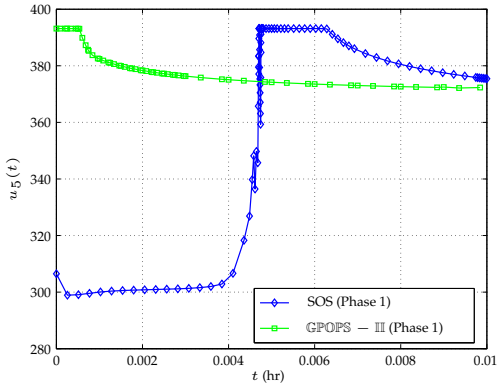
(b) $u_2(t)$ vs. t.



(c) $u_3(t)$ vs. t.



(d) $u_4(t)$ vs. t.



(e) $u_5(t)$ vs. t.

Fig. 17: Phase 1 Control Solution to Kinetic Batch Reactor Optimal Control Problem Using GPOPS − II Alongside Solution Obtained Using Optimal Control Software *Sparse Optimization Suite*.

5.5. Multiple-Stage Launch Vehicle Ascent Problem

The problem considered in this section is the ascent of a multiple-stage launch vehicle. The objective is to maneuver the launch vehicle from the ground to the target orbit while maximizing the remaining fuel in the upper stage. It is noted that this example is is found verbatim in Benson [2004], Rao et al. [2010], and Betts [2010]. The problem is modeled using four phases where the objective is to maximize the mass at the end of the fourth phase, that is maximize

$$J = m(t_f^{(4)}) \tag{82}$$

subject to the dynamic constraints

$$\begin{aligned}
\dot{\mathbf{r}}^{(p)} &= \mathbf{v}^{(p)}, \\
\dot{\mathbf{v}}^{(p)} &= -\frac{\mu}{\|\mathbf{r}^{(p)}\|^3}\mathbf{r}^{(p)} + \frac{T^{(p)}}{m^{(p)}}\mathbf{u}^{(p)} + \frac{\mathbf{D}^{(p)}}{m^{(p)}}, \quad (p = 1, \ldots, 4), \\
\dot{m}^{(p)} &= -\frac{T^{(p)}}{g_0 I_{sp}},
\end{aligned} \tag{83}$$

the initial conditions

$$\begin{aligned}
\mathbf{r}(t_0) &= \mathbf{r}_0 = (5605.2, 0, 3043.4) \times 10^3 \text{ m}, \\
\mathbf{v}(t_0) &= \mathbf{v}_0 = (0, 0.4076, 0) \times 10^3 \text{ m/s}, \\
m(t_0) &= m_0 = 301454 \text{ kg}.
\end{aligned} \tag{84}$$

the interior point constraints

$$\begin{aligned}
\mathbf{r}^{(p)}(t_f^{(p)}) - \mathbf{r}^{(p+1)}(t_0^{(p+1)}) &= \mathbf{0}, \\
\mathbf{v}^{(p)}(t_f^{(p)}) - \mathbf{v}^{(p+1)}(t_0^{(p+1)}) &= \mathbf{0}, \qquad (p = 1, \ldots, 3) \\
m^{(p)}(t_f^{(p)}) - m_{\text{dry}}^{(p)} - m^{(p+1)}(t_0^{(p+1)}) &= 0,
\end{aligned} \tag{85}$$

the terminal constraints (corresponding to a geosynchronous transfer orbit),

$$\begin{aligned}
a(t_f^{(4)}) &= a_f = 24361.14 \text{ km} \quad , \quad e(t_f^{(4)}) = e_f = 0.7308, \\
i(t_f^{(4)}) &= i_f = 28.5 \deg \quad , \quad \theta(t_f^{(4)}) = \theta_f = 269.8 \deg, \\
&\phi(t_f^{(4)}) = \phi_f = 130.5 \deg,
\end{aligned} \tag{86}$$

and the path constraints

$$\begin{aligned}
\|\mathbf{r}^{(p)}\|_2^2 &\geq R_e, \\
\|\mathbf{u}^{(p)}\|_2^2 &= 1,
\end{aligned} \quad (p = 1, \ldots, 4). \tag{87}$$

In each phase $\mathbf{r}(t) = (x(t), y(t), z(t))$ is the position relative to the center of the Earth expressed in ECI coordinates, $\mathbf{v} = (v_x(t), v_y(t), v_z(t))$ is the inertial velocity expressed in ECI coordinates, μ is the gravitational parameter, T is the vacuum thrust, m is the mass, g_0 is the acceleration due to gravity at sea level, I_{sp} is the specific impulse of the engine, $\mathbf{u} = (u_x, u_y, u_z)$ is the thrust direction expressed in ECI coordinates, and $\mathbf{D} = (D_x, D_y, D_z)$ is the drag force expressed ECI coordinates. The drag force is defined as

$$\mathbf{D} = -\tfrac{1}{2} C_D S \rho \|\mathbf{v}_{\text{rel}}\| \mathbf{v}_{\text{rel}} \tag{88}$$

where C_D is the drag coefficient, S is the vehicle reference area, $\rho = \rho_0 \exp(-h/H)$ is the atmospheric density, ρ_0 is the sea level density, $h = r - R_e$ is the altitude, $r = \|\mathbf{r}\|_2 = \sqrt{x^2 + y^2 + z^2}$ is the geocentric radius, R_e is the equatorial radius of the Earth, H is the

density scale height, and $\mathbf{v}_{\text{rel}} = \mathbf{v} - \boldsymbol{\omega} \times \mathbf{r}$ is the velocity as viewed by an observer fixed to the Earth expressed in ECI coordinates, and $\boldsymbol{\omega} = (0, 0, \Omega)$ is the angular velocity of the Earth as viewed by an observer in the inertial reference frame expressed in ECI coordinates. Furthermore, m_{dry} is the dry mass of phases 1, 2, and 3 and is defined $m_{\text{dry}} = m_{\text{tot}} - m_{\text{prop}}$, where m_{tot} and m_{prop} are, respectively, the total mass and dry mass of phases 1, 2, and 3. Finally, the quantities a, e, i, θ, and ϕ are, respectively, the semi-major axis, eccentricity, inclination, longitude of ascending node, and argument of periapsis, respectively. The vehicle data for this problem and the he numerical values for the physical constants can be found in Tables III. and IV, respectively.

Table III: Vehicle Properties for Multiple-Stage Launch Vehicle Ascent Problem.

Quantity	Solid Boosters	Stage 1	Stage 2
m_{tot} (kg)	19290	104380	19300
m_{prop} (kg)	17010	95550	16820
T (N)	628500	1083100	110094
I_{sp} (s)	283.3	301.7	467.2
Number of Engines	9	1	1
Burn Time (s)	75.2	261	700

Table IV: Constants Used in the Launch Vehicle Ascent Optimal Control Problem.

Constant	Value
Payload Mass	4164 kg
S	4π m^2
C_D	0.5
ρ_0	1.225 kg/m^3
H	7200 m
t_1	75.2 s
t_2	150.4 s
t_3	261 s
R_e	6378145 m
Ω	$7.29211585 \times 10^{-5}$ rad/s
μ	3.986012×10^{14} m^3/s^2
g_0	9.80665 m/s^2

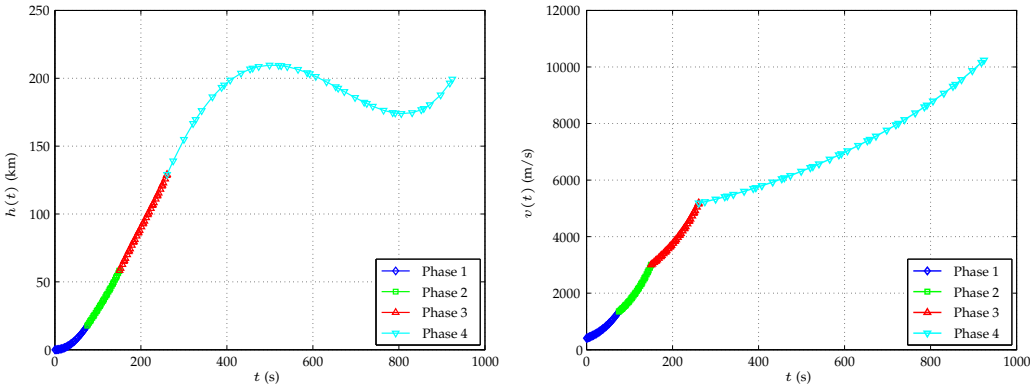
The multiple-stage launch vehicle ascent optimal control problem was solved using $\mathbb{GPOPS}-\mathbb{II}$ with the NLP solver SNOPT and an initial mesh in each phase consisting of ten uniformly spaced mesh intervals with four LGR points per mesh interval, and

the following initial guess:

$$
\begin{aligned}
\mathbf{r}^{(1)} &= \mathbf{r}(0),\\
\mathbf{v}^{(1)} &= \mathbf{v}(0),\\
m^{(1)} &= \mathbf{linear}(t_0^{(1)}, t_f^{(1)}, 3.01454, 1.71863) \times 10^5 \text{ kg},\\
\mathbf{u}^{(1)} &= (0,1,0),\\
t_0^{(1)} &= 0,\\
t_f^{(1)} &= 75.2 \text{ s},
\end{aligned}
\qquad
\begin{aligned}
\mathbf{r}^{(2)} &= \mathbf{r}(0),\\
\mathbf{v}^{(2)} &= \mathbf{v}(0),\\
m^{(2)} &= \mathbf{linear}(t_0^{(2)}, t_f^{(2)}, 1.58184, 0.796238) \times 10^5 \text{ kg},\\
\mathbf{u}^{(2)} &= (0,1,0),\\
t_0^{(2)} &= 75.2 \text{ s},\\
t_f^{(2)} &= 150.4 \text{ s},
\end{aligned}
$$

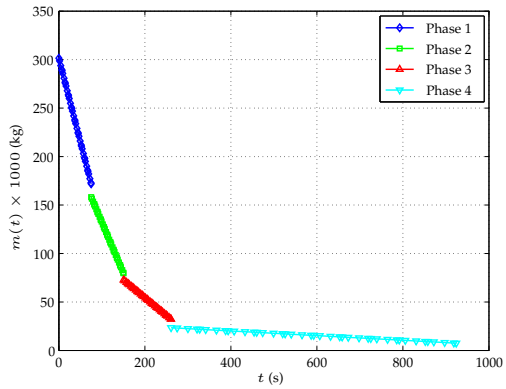
$$
\begin{aligned}
\mathbf{r}^{(3)} &= \bar{\mathbf{r}},\\
\mathbf{v}^{(3)} &= \bar{\mathbf{v}},\\
m^{(3)} &= \mathbf{linear}(t_0^{(3)}, t_f^{(3)}, 7.27838, 3.22940) \times 10^4 \text{ kg},\\
\mathbf{u}^{(3)} &= (0,1,0),\\
t_0^{(3)} &= 150.4 \text{ s},\\
t_f^{(3)} &= 261 \text{ s},
\end{aligned}
\qquad
\begin{aligned}
\mathbf{r}^{(4)} &= \bar{\mathbf{r}},\\
\mathbf{v}^{(4)} &= \bar{\mathbf{v}},\\
m^{(4)} &= \mathbf{linear}(t_0^{(4)}, t_f^{(4)}, 2.34640, 0.41640) \times 10^4 \text{ kg},\\
\mathbf{u}^{(4)} &= (0,1,0),\\
t_0^{(4)} &= 261 \text{ s},\\
t_f^{(4)} &= 961 \text{ s},
\end{aligned}
$$

where $(\bar{\mathbf{r}}, \bar{\mathbf{v}})$ are obtained from a conversion of the orbital elements $(a, e, i, \theta, \phi, \nu) = (a_f, e_f, i_f, \theta_f, \phi_f, 0)$ to ECI Cartesian coordinates [Bate et al. 1971] and ν is the true anomaly. The $\mathbb{GPOPS}-\mathbb{II}$ solution is shown in Fig. 18. In this example the mesh refinement accuracy tolerance of 10^{-6} is satisfied on the initial mesh and, thus, no mesh refinement is performed. The solution obtained using $\mathbb{GPOPS}-\mathbb{II}$ matches closely with the solution obtained using the software *SOCS* [Betts 2010], where it is noted that the optimal objective values obtained using $\mathbb{GPOPS}-\mathbb{II}$ and SOCS are -7529.712680 and -7529.712412, respectively.

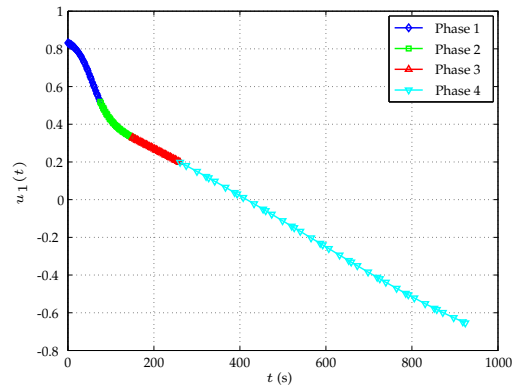


(a) $h(t)$ vs. t.

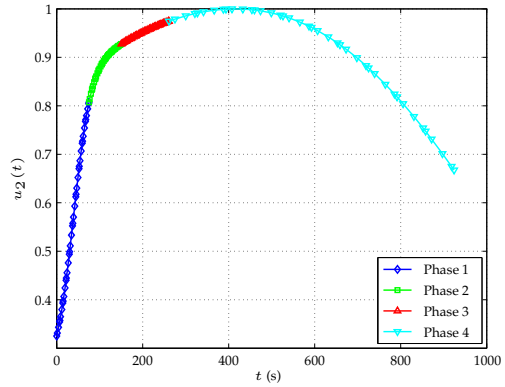
(b) $v(t)$ vs. t.



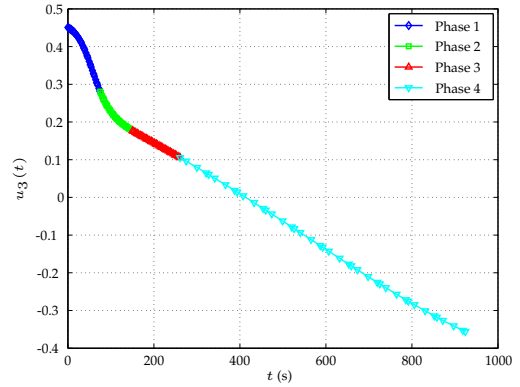
(c) $m(t)$ vs. t.



(d) $u_1(t)$ vs. t.



(e) $u_2(t)$ vs. t.



(f) $u_3(t)$ vs. t.

Fig. 18: Solution of Multiple-Stage Launch Vehicle Ascent Problem Using GPOPS − II.

6. CAPABILITIES OF $\mathbb{GPOPS} - \mathbb{II}$

The five examples provide in Section 5 highlight the capability of $\mathbb{GPOPS} - \mathbb{II}$ to solve a wide variety of problems. First, the hyper-sensitive example shows the capability of the ph mesh refinement method in increasing the number of mesh intervals and collocation points in regions where the dynamics are changing more rapidly, while the reusable launch vehicle entry example demonstrates the ability of the ph mesh refinement method to keep the number of collocation points much smaller than might be possible using a fixed low-order collocation method. Second, the space station attitude control and kinetic batch reactor examples demonstrate the flexibility of the software to achieve better performance on an application by modifying the default settings. Third, the launch vehicle ascent problem shows that $\mathbb{GPOPS} - \mathbb{II}$ has been designed with the ability to employ different NLP solvers. The different examples also demonstrate the wide variety of problems that can be formulated in $\mathbb{GPOPS} - \mathbb{II}$. Such problems range from one-phase problems with a Lagrange cost (for example, the hyper-sensitive and space station attitude control examples), Mayer cost (for example, the reusable launch vehicle entry and launch vehicle ascent examples), and problems that include static parameters (for example, the kinetic batch reactor example). Moreover, it was shown that $\mathbb{GPOPS} - \mathbb{II}$ has the ability to solve challenging multiple-phase problems (for example, the kinetic batch reactor example). The fact that $\mathbb{GPOPS} - \mathbb{II}$ is capable of solving the challenging benchmark optimal control problems shown in this paper shows the general utility of the software on problems that may arise in different application areas.

7. LIMITATIONS OF $\mathbb{GPOPS} - \mathbb{II}$

Like all software, $\mathbb{GPOPS} - \mathbb{II}$ has limitations. First, it is assumed in the implementation that all functions have continuous first and second derivatives. In some applications, however, the functions themselves may be continuous while the derivatives may be discontinuous. In such problems $\mathbb{GPOPS} - \mathbb{II}$ may struggle because the NLP solver is not being provided with accurate approximations to the derivative functions. Furthermore, the ability of any given NLP solver to obtain a solution is always problem dependent. As a result, for some examples it may be the case that IPOPT will perform better than SNOPT, but in some cases SNOPT may may significantly outperform IPOPT (the launch vehicle ascent problem is an example where SNOPT outperforms IPOPT with $\mathbb{GPOPS} - \mathbb{II}$). Also, problems with high-index path constraints may result in the constraint qualification conditions not being satisfied on fine meshes. In such cases, unique NLP Lagrange multipliers may not exist. In some cases, these Lagrange multipliers may become unbounded. Finally, as is true for many optimal control software programs, applications whose solutions lie on a singular arc can create problems due to the inability to determine the optimal control along the singular arc. In such cases highly inaccurate solution may be obtained in the region near the singular arc, and mesh refinement may only exacerbate these inaccuracies. The approach for problems whose solutions lie on a singular arc is to modify the problem by including the conditions that define the singular arc (thus removing the singularity).

8. CONCLUSIONS

A general-purpose MATLAB software program called $\mathbb{GPOPS} - \mathbb{II}$ has been described for solving multiple-phase optimal control problems using variable-order Gaussian quadrature methods. In particular, the software employs a Legendre-Gauss-Radau quadrature orthogonal collocation where the continuous-time control problem is transcribed to a large sparse nonlinear programming problem. The software implements two previously developed adaptive mesh refinement methods that allow for flexibility

in the number and placement of the collocation and mesh points in order to achieve a specified accuracy. In addition, the software is designed to compute all derivatives required by the NLP solver using sparse finite-differencing of the optimal control functions. The key components of the software have been described in detail and the utility of the software is demonstrated on five benchmark optimal control problems. The software described in this paper provides researchers a useful platform upon which to solve a wide variety of complex constrained optimal control problems.

Acknowledgments

The authors gratefully acknowledge support for this research from the U.S. Office of Naval Research under Grant N00014-11-1-0068 and from the U.S. Defense Advanced Research Projects Agency under Contract HR0011-12-C-0011.

Disclaimer

The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

REFERENCES

ABRAMOWITZ, M. AND STEGUN, I. 1965. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, New York.

BABUSKA, I. AND RHEINBOLDT, W. C. 1979. Reliable error estimation and mesh adaptation for the finite element method. In *Proc. Second Intl. Conf. Computational Methods in Nonlinear Mechanics (Univ. Texas Austin)*. North-Holland, Amsterdam-New York, 67–108.

BABUSKA, I. AND RHEINBOLDT, W. C. 1981. A posteriori error analysis of finite element solutions for one-dimensional problems. *SIAM Journal on Numerical Analysis 18*, 565–589.

BABUSKA, I. AND RHEINBOLDT, W. C. 1982. Computational error estimates and adaptive processes for some nonlinear structural problems. *Computer Methods in Applied Mechanics and Engineering 34,* 1–3, 895–937.

BABUSKA, I., ZIENKIEWICZ, O. C., GAGO, J., AND DE A. OLIVEIRA, E. R. 1986. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*. John Wiley & Sons, Chichester.

BATE, R. R., MUELLER, D. D., AND WHITE, J. E. 1971. *Fundamentals of Astrodynamics*. Dover Publications, Mineola, New York.

BECERRA, V. M. 2009. *PSOPT Optimal Control Solver User Manual*. University of Reading. http://www.psopt.org.

BENSON, D. A. 2004. A Gauss pseudospectral transcription for optimal control. Ph.D. thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts.

BENSON, D. A., HUNTINGTON, G. T., THORVALDSEN, T. P., AND RAO, A. V. 2006. Direct trajectory optimization and costate estimation via an orthogonal collocation method. *Journal of Guidance, Control, and Dynamics 29,* 6, 1435–1440.

BETTS, J. T. 1990. Sparse jacobian updates in the collocation method for optimal control problems. *Journal of Guidance, Control, and Dynamics 13,* 3, 409–415.

BETTS, J. T. 1998. Survey of numerical methods for trajectory optimization. *Journal of Guidnance, Control, and Dynamics 21,* 2, 193–207.

BETTS, J. T. 2010. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. SIAM Press, Philadelphia.

BETTS, J. T. 2013. Sparse Optimization Suite (SOS). Applied Mathematical Analysis, LLC, Based on the Algorithms Published in Betts, J. T., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, SIAM Press, Philadelphia, 2010.

BIEGLER, L. T., GHATTAS, O., HEINKENSCHLOSS, M., AND VAN BLOEMEN WAANDERS, B., Eds. 2003. *Large-Scale PDE Constrained Optimization*. Lecture Notes in Computational Science and Engineering, Vol. 30. Springer-Verlag, Berlin.

BIEGLER, L. T. AND ZAVALA, V. M. 2008. Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide optimization. *Computers and Chemical Engineering 33,* 3, 575–582.

BYRD, R. H., NOCEDAL, J., AND WALTZ, R. A. 2006. Knitro: An integrated package for nonlinear optimization. In *Large Scale Nonlinear Optimization*. Springer Verlag, 35–59.

CANUTO, C., HUSSAINI, M. Y., QUARTERONI, A., AND ZANG, T. A. 1988. *Spectral Methods in Fluid Dynamics*. Spinger-Verlag, Heidelberg, Germany.

DARBY, C. L., GARG, D., AND RAO, A. V. 2011a. Costate estimation using multiple-interval pseudospectral methods. *Journal of Spacecraft and Rockets 48,* 5, 856–866.

DARBY, C. L., HAGER, W. W., AND RAO, A. V. 2011b. Direct trajectory optimization using a variable low-order adaptive pseudospectral method. *Journal of Spacecraft and Rockets 48,* 3, 433–445.

DARBY, C. L., HAGER, W. W., AND RAO, A. V. 2011c. An hp-adaptive pseudospectral method for solving optimal control problems. *Optimal Control Applications and Methods 32,* 4, 476–502.

ELNAGAR, G., KAZEMI, M., AND RAZZAGHI, M. 1995. The pseudospectral Legendre method for discretizing optimal control problems. *IEEE Transactions on Automatic Control 40,* 10, 1793–1796.

ELNAGAR, G. AND RAZZAGHI, M. 1998. A collocation-type method for linear quadratic optimal control problems. *Optimal Control Applications and Methods 18,* 3, 227–235.

FALUGI, P., KERRIGAN, E., AND VAN WYK, E. 2010. *Imperial College London Optimal Control Software User Guide (ICLOCS)*. Department of Electrical Engineering, Imperial College London, London, UK.

FORNBERG, B. 1998. *A Practical Guide to Pseudospectral Methods*. Cambridge University Press, New York.

GARG, D., HAGER, W. W., AND RAO, A. V. 2011a. Pseudospectral methods for solving infinite-horizon optimal control problems. *Automatica 47,* 4, 829–837.

GARG, D., PATTERSON, M., FRANCOLIN, C., DARBY, C., HUNTINGTON, G., HAGER, W. W., AND RAO, A. V. 2011b. Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a Radau pseudospectral method. *Computational Optimization and Applications 49,* 2, 335–358.

GARG, D., PATTERSON, M., HAGER, W. W., RAO, A. V., BENSON, D. A., AND HUNTINGTON, G. T. 2010. A unified framework for the numerical solution of optimal control problems using pseudospectral methods. *Automatica 46,* 11, 1843–1851.

GILL, P. E., MURRAY, W., AND SAUNDERS, M. A. 2002. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review 47,* 1, 99–131.

GILL, P. E., MURRAY, W., AND WRIGHT, M. H. 1981. *Practical Optimization*. Academic Press, London.

GOH, C. J. AND TEO, K. L. 1988. Control parameterization: A unified approach to optimal control problems with general constraints. *Automatica 24,* 1, 3–18.

GONG, Q., FAHROO, F., AND ROSS, I. M. 2008a. Spectral algorithm for pseudospectral methods in optimal control. *Journal of Guidance, Control and Dynamics 31,* 3, 460–471.

GONG, Q., ROSS, I. M., KANG, W., AND FAHROO, F. 2008b. Connections between the covector mapping theorem and convergence of pseudospectral methods. *Computational Optimization and Applications 41,* 3, 307–335.

HOUSKA, B., FERREAU, H. J., AND DIEHL, M. 2011. ACADO toolkit – an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods 32,* 3, 298–312.

HUNTINGTON, G. T., BENSON, D. A., AND RAO, A. V. 2007. Optimal configuration of tetrahedral spacecraft formations. *The Journal of the Astronautical Sciences 55,* 2, 141–169.

HUNTINGTON, G. T. AND RAO, A. V. 2008. Optimal reconfiguration of spacecraft formations using the Gauss pseudospectral method. *Journal of Guidance, Control, and Dynamics 31,* 3, 689–698.

JAIN, D. AND TSIOTRAS, P. 2008. Trajectory optimization using multiresolution techniques. *Journal of Guidance, Control, and Dynamics 31,* 5, 1424–1436.

JANSCH, C., WELL, K. H., AND SCHNEPPER, K. 1994. *GESOP - Eine Software Umgebung Zur Simulation Und Optimierung*. Proceedings des SFB.

KAMESWARAN, S. AND BIEGLER, L. T. 2008. Convergence rates for direct transcription of optimal control problems using collocation at Radau points. *Computational Optimization and Applications 41,* 1, 81–126.

LEINEWEBER, D. B. 1998. Efficient reduced SQP methods for the optimization of chemical processes described by large space dae models. Ph.D. thesis, Universität Heidelberg, Interdisziplinäres Zentrum für Wissenschaftliches Rechnen (IWR).

MUMPS. 2011. Multifrontal massively parallel solver (mumps 4.10.0) user's guide.

PATTERSON, M. A. AND RAO, A. V. 2012. Exploiting sparsity in direct collocation pseudospectral methods for solving optimal control problems. *Journal of Spacecraft and Rockets 49,* 2, 364–377.

PATTERSON, M. A., RAO, A. V., AND HAGER, W. W. 2013. A ph collocation scheme for optimal control. *Optimal Control Applications and Methods*, In Revision.

PIETZ, J. A. 2003. Pseudospectral collocation methods for the direct transcription of optimal control problems. M.S. thesis, Rice University, Houston, Texas.

ÅKESSON, J., ARZEN, K. E., GÄFERT, M., BERGDAHL, T., AND TUMMESCHEIT, H. 2010. Modeling and Optimization with Optimica and JModelica.org – Languages and Tools for Solving Large-Scale Dynamic Optimization Problems. *Computers and Chemical Engineering 34,* 11, 1737–1749.

RAO, A. V., BENSON, D. A., DARBY, C., PATTERSON, M. A., FRANCOLIN, C., SANDERS, I., AND HUNTINGTON, G. T. 2010. Algorithm 902: GPOPS, A MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method. *ACM Transactions on Mathematical Software 37,* 2, 22:1–22:39.

RAO, A. V. AND MEASE, K. D. 2000. Eigenvector approximate dichotomic basis method for solving hypersensitive optimal control problems. *Optimal Control Applications and Methods 21,* 1, 1–19.

TREFETHEN, L. N. 2000. *Spectral Methods Using MATLAB*. SIAM Press, Philadelphia.

VLASES, W. G., PARIS, S. W., LAJOIE, R. M., MARTENS, M. J., AND HARGRAVES, C. R. 1990. Optimal trajectories by implicit simulation. Tech. Rep. WRDC-TR-90-3056, Boeing Aerospace and Electronics, Wright-Patterson Air Force Base, Ohio.

VON STRYK, O. 2000. *User's Guide for DIRCOL (Version 2.1): A Direct Collocation Method for the Numerical Solution of Optimal Control Problems*. Technical University, Munich, Germany.

WÄCHTER, A. AND BIEGLER, L. T. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming 106,* 1, 25–57.

ZHAO, Y. AND TSIOTRAS, P. 2011. Density functions for mesh refinement in numerical optimal control. *Journal of Guidance, Control, and Dynamics 34,* 1, 271–277.