2008-08-18

Guess-Free Trajectory Optimization

Gong, Qi

The American Institute of Aeronautics and Astronautics (AIAA)

AIAA/AAS Astrodynamics Specialist Conference and Exhibit
18 - 21 August 2008, Honolulu, Hawaii

AIAA 2008-6273

Guess-Free Trajectory Optimization

I. Michael Ross* and Qi Gong†

Trajectory optimization is an open-loop optimal control problem. The necessary conditions for optimal control are easy to generate by way of Pontryagin's Principle. Although these necessary conditions are powerful analysis tools, the "curse of complexity" has long been a major obstacle to solving the resulting boundary value problem. The Covector Mapping Principle overcomes this curse by implying that an optimal control problem is solvable if dualization is commuted with discretization. This solvability is predicated through two notions of convergence: one related to the solution and another to the algorithm. We design an elastic programming technique to exploit the differences between these two notions of convergence while harnessing their coupling to propel a spectral algorithm. The combination of these concepts generates a globally convergent algorithm under mild conditions. In meeting the mild assumptions required of this guess-free optimization technique, the problem must be reasonably bounded and reasonably scaled. Both these requirements can be easily met for a very large family of practical problems in engineering through a new computational interpretation of the adjoints that provides useful equations for balancing the primal problem.

I. Introduction

One of the main problems in the theory, design and implementation of any algorithm is its starting point or guess. Roughly speaking, the starting point problem can be stated as follows: if the guess or the starting point is close to the optimal solution, the algorithm converges; if it is far, the algorithm may not converge. One of the most basic mathematical proofs of this convergence result is the Newton-Kantorovich Theroem.[1,2] Over the last two decades, there has been significant progress within the optimization community in globalization techniques:[3,4] that is, techniques that facilitate convergence of the algorithm from arbitrary starting points. Such an algorithm is said to be globally convergent. In trajectory optimization, yet another notion of convergence is required. This is the notion of solution convergence[5] – that is, the requirement that the computed solution converge to the optimal solution. The absence of this notion of convergence or its demonstration in practical problems may lead some practitioners to false claims of optimality.

Historically, trajectory optimization problems have had convergence problems because a number of additional issues, the most famous of these being Bellman's "curse of dimensionality." Over the last decade, significant strides in trajectory optimization has been made, in part by major advances in pseudospectral (PS) methods.[6] The flight implementation of PS methods[7–9] for maneuvering the International Space Station illustrates the high confidence that operators and practitioners have bestowed on its demonstrable convergence properties. In certain applications, PS methods already exhibit global convergence properties. For instance, in Ref. [10], Ross and Fahroo showed that PS methods converge to the optimal solution for various formulations of a robotics problem from randomly chosen starting points. Emboldened by such numerical studies, we describe and demonstrate some new ideas that have the effect of eliminating the guessing problem in trajectory optimization. Our proposed algorithm integrates a number of recent advances that have occurred in both mathematical programming techniques and PS methods. In particular, an elastic programming concept is incorporated as part of the spectral algorithm to push an arbitrary starting point into a region that is closer to the optimal point. Under a reasonable bound assumption (explained in Sec. IV), the algorithm displays the properties of robustness with respect to the starting point. This robustness property is exploited to design a guess-free trajectory optimization algorithm.

*Professor and Program Director, Control & Optimization, Naval Postgraduate School, Monterey, CA 93943; imross@nps.edu. Associate Fellow, AIAA. `http://www.nps.edu/academics/gnclab/`

†Assistant Professor, Department of Applied Mathematics and Statistics, University of California, Santa Cruz, CA, 95064. E-mail: `qigong@soe.ucsc.edu`.

It is important to note the guess-free spectral algorithm is applicable only over the "space" of trajectory optimization problems and not over the space of all optimization problems. That is, trajectory optimization problems are not merely large-scale optimization problems; the continuity of time and the structure of the constraints can be used to tailor optimization algorithms to solve trajectory problems. In PS methods, the discrete optimization problem preserves the algebraic and geometric structure of the continuous-time problem[10] and this specific structure can be exploited to good use in conjunction with the relatively low-scale optimization problem produced by PS discretization. Thus, by tailoring and adapting globalization techniques developed in the optimization community to a class of PS-structured optimization problems, we achieve robustness, and hence a guess-free trajectory optimization algorithm.

II. The Curse of Complexity

Trajectory optimization problems are essentially open-loop optimal control problems. A basic open-loop optimal control problem can be defined as follows: Find the state-control function pair, $t \mapsto (\boldsymbol{x}, \boldsymbol{u}) \in (\mathbb{R}^{N_x} \times \mathbb{R}^{N_u})$, and clock times, t_0, and t_f, that

$$(\mathsf{B}) \left\{ \begin{array}{ll} \textsf{Minimize} & J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), t_0, t_f] \\ & \quad = E(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) + \displaystyle\int_{t_0}^{t_f} F(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \\ \textsf{Subject to} & \dot{\boldsymbol{x}}(t) = \mathbf{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\ & \mathbf{e}\big(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f\big) = \mathbf{0} \\ & \mathbf{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \le \mathbf{0} \end{array} \right.$$

where, $\boldsymbol{x}_0 = \boldsymbol{x}(t_0), \boldsymbol{x}_f = \boldsymbol{x}(t_f)$, and all the relevant relationships are assumed to be true for almost all t. It is assumed that all the nonlinear functions, F, E, $\mathbf{f}$, $\mathbf{e}$, and $\mathbf{h}$, are continuously differentiable with respect to their arguments and that their gradients are Lipschitz continuous over the domain.

The necessary conditions for Problem B can be articulated in terms of a boundary value problem which is, more formally, a problem of solving a generalized equation.[11] We refer to this process as dualization, and it can be summarized[12] as Problem B^λ:

$$(\mathsf{B}^\lambda) \left\{ \begin{array}{ll} \textsf{Find} & [\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), t_0, t_f; \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}(\cdot), \boldsymbol{\nu}] \\ \textsf{Such that} & \dot{\boldsymbol{x}}(t) = \mathbf{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\ & \mathbf{e}\big(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f\big) = \mathbf{0} \\ & \mathbf{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \le \mathbf{0} \\ & \dot{\boldsymbol{\lambda}}(t) = -\dfrac{\partial \overline{H}[t]}{\partial \boldsymbol{x}} \\ & \dfrac{\partial \overline{H}}{\partial \boldsymbol{u}} = 0 \\ & \{\boldsymbol{\lambda}(t_0), \boldsymbol{\lambda}(t_f)\} = \left\{ -\dfrac{\partial \overline{E}}{\partial \boldsymbol{x}_0}, \dfrac{\partial \overline{E}}{\partial \boldsymbol{x}_f} \right\} \\ & \{H[t_0], H[t_f]\} = \left\{ \dfrac{\partial \overline{E}}{\partial t_0}, -\dfrac{\partial \overline{E}}{\partial t_f} \right\} \\ & \mathbf{0} \le \boldsymbol{\mu}(t) \perp - \mathbf{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \ge \mathbf{0} \end{array} \right.$$

where $\overline{H}$ is the Lagrangian of the Hamiltonian,

$$\overline{H}(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{x}, \boldsymbol{u}) := H(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\mu}^T \mathbf{h}(\boldsymbol{x}, \boldsymbol{u}) \tag{1}$$

H is the control Hamiltonian,

$$H(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}) := F(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T \mathbf{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2}$$

$\overline{E}$ is the endpoint Lagrangian,

$$\overline{E}(\boldsymbol{\nu}, \boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) := E(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) + \boldsymbol{\nu}^T \mathbf{e}(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) \tag{3}$$

and the shorthand notation, $\overline{H}[t_f]$ is used to imply,

$$\overline{H}[t_f] \equiv H(\boldsymbol{\lambda}(t_f), \boldsymbol{\mu}(t_f), \boldsymbol{x}(t_f), \boldsymbol{u}(t_f))$$

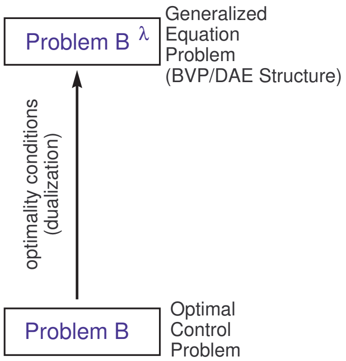
Problem B $^{\lambda}$
Generalized Equation Problem (BVP/DAE Structure)
optimality conditions (dualization)
Problem B
Optimal Control Problem

Figure 1. Pontryagin lift of an optimal control problem.

This so-called Pontryagin "Lift" is illustrated in Fig. 1. The premise of using the necessary conditions as a problem solving tool is that it is easier to solve a "larger" problem (namely Problem B^{λ}) in the primal-dual space than the "smaller" primal problem. This is somewhat true from an analytical point of view; however, solving an open-loop optimal control problem by numerically solving for the necessary conditions — sometimes called an "indirect" method — has well-known difficulties[13] that can be encapsulated as[14] the "curse of complexity." Similar to the curse of dimensionality,[15] this problem is fundamental and not technological as the following example[14] illustrates:

$$\boldsymbol{x} = [x] \quad \boldsymbol{u} = [u] \qquad \mathbb{U} = \mathbb{R}$$

$$\left\{ \begin{array}{lrl} \text{Minimize} & J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)] = & \displaystyle\int_0^{t_f} \frac{u^2}{2}\, dt \\ \text{Subject to} & \dot{x}(t) = & ax(t) \\ & x_0 = & x^0 \\ & t_f = & 1 \end{array} \right. \tag{4}$$

From Pontryagin's Principle, the adjoint equation is $\dot{\lambda} = -a\lambda$; hence, we have a system of differential equations,

$$\begin{pmatrix} \dot{x} \\ \dot{\lambda} \end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & -a \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} \tag{5}$$

that constitutes a Hamiltonian system. These equations form an unstable system as it has one pole in the right-half plane (and another one in the left-half plane). Thus, if the state dynamics is dissipative (i.e., $a < 0$), the adjoint system exhibits an exponential growth. Conversely, if the adjoint system is stable (i.e. $a > 0$) then the state is unstable.

The exact solution of the state-costate pair is given by,

$$\phi(t; t_0, x_0) \quad = \quad x_0 e^{a(t-t_0)} \tag{6}$$

$$\psi(t; t_0, \lambda_0) \quad = \quad \lambda_0 e^{-a(t-t_0)} \tag{7}$$

Replacing x_0 by $x^0 + \varepsilon_x$ and λ_0 by $\lambda^0 + \varepsilon_\lambda$, we get the *exact solution* for the perturbed system,

$$\phi_\varepsilon(t; t_0, \varepsilon_x) \quad = \quad (x^0 + \varepsilon_x) e^{a(t-t_0)} \tag{8}$$

$$\psi_\varepsilon(t; t_0, \varepsilon_\lambda) \quad = \quad (\lambda^0 + \varepsilon_\lambda) e^{-a(t-t_0)} \tag{9}$$

Now suppose we had an error in just the initial condition but somehow managed to get an exact propagation; then, the errors would propagate as,

$$\begin{aligned} \delta x := \phi_\varepsilon(t; t_0, \varepsilon_x) - \phi_\varepsilon(t; t_0, 0) \quad &= \varepsilon_x e^{a(t-t_0)} \\ \delta\lambda := \psi_\varepsilon(t; t_0, \varepsilon_\lambda) - \psi_\varepsilon(t; t_0, 0) \quad &= \varepsilon_\lambda e^{-a(t-t_0)} \end{aligned} \tag{10}$$

Thus, if $a < 0$, a small error in the initial condition in the state decreases exponentially in time (good news) while the corresponding error in the costate increases (bad news). On the other hand, if $a > 0$, the situation is reversed. Thus, it is impossible to simultaneously control the error growth in the state-costate pair. This difficulty is more revealing from the observation,

$$\delta x \, \delta\lambda = \varepsilon_x \, \varepsilon_\lambda$$

which indicates that the "error area" is preserved under *perfect integration*. Thus, reducing one side of the error rectangle ensures the elongation of the other − this is the well-known property associated with the symplectic structure of all Hamiltonian systems. Note that this analysis is independent of any numerical propagation scheme. A practical numerical integration scheme on a digital computer would guarantee this problem. This is because even if the values of x^0 and λ^0 were known exactly, a numerical propagation scheme would automatically introduce ε_x and ε_λ in the immediate next step of the integration cycle as a result of machine errors (round-off) and truncation errors (in the integration scheme). Thus, the reality is actually worse than Eq.(10), and then even more so, because numerical propagation errors are cumulative.

It is apparent that the difficulties encountered in solving the Hamiltonian system (such as Eq. (5)) are a direct consequence of its symplectic structure. This is why solving a Hamiltonian boundary value problem has an inherent "curse of complexity" that cannot be overcome.[14] We hasten to note that while symplectic integrators[16] avoid this problem in principle, they are not universal. That is, a proper application of symplectic integrators requires a careful analysis of the poles of the system, and these poles are not guaranteed to stay on one side of the plane (e.g. left side) as the integration proceeds. This is why the curse of complexity remains as a general principle, and is illustrated in Fig. 2.

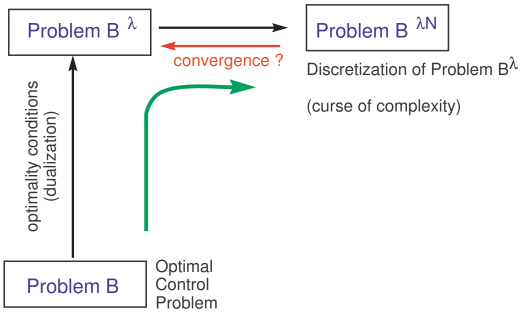

Figure 2. The curse of complexity in an "indirect" method.

III. Pseudospectral Methods

Solving optimal control problems using approximation methods goes back to the days of Bernoulli and Euler and the founding of optimal control theory itself.[17–20] In this context, PS methods can be described as modernized versions of the "lost" principles originally laid our by Bernoulli and Euler.

A. Approximation of Functions, Derivatives and Integrals

The key to our proposed theory is the use of weighted interpolants of the form,[6,21,22]

$$y(t) \approx y^N(t) = \sum_{j=0}^{N} \frac{W(t)}{W(t_j)} \phi_j(t) y_j, \;\; a \le t \le b \tag{11}$$

where $y(t)$ is an arbitrary function. Here the nodes $t_j, j = 0, ..., N$ are a set of distinct interpolation nodes (defined later) on the interval $[a, b]$, the weight function $W(t)$ is a positive function on the interval, and $\phi_j(t)$ is the $Nth-$ order Lagrange interpolating polynomial that satisfies the relationship $\phi_j(t_k) = \delta_{jk}$. This implies that

$$y_j = y^N(t_j), \;\; j = 0, ...N. \tag{12}$$

An expression for the Lagrange polynomial can be written as[23]

$$\phi_j(t) = \frac{g_N(t)}{g'_N(t_j)(t - t_j)}, \;\; g_N(t) = \prod_{j=0}^{N}(t - t_j). \tag{13}$$

One important tenant of PS approximation of functions is that differentiation of the approximated functions can be performed by differentiation of the interpolating polynomial,

$$\frac{dy^N(t)}{dt} = \sum_{j=0}^{N} \frac{y_j}{W(t_j)}[W'(t)\phi_j(t) + W(t)\phi_j']$$

Since only the values of the derivative at the nodes t_i are required for PS methods, then we have,

$$\left.\frac{dy^N(t)}{dt}\right|_{t_i} = \sum_{j=0}^{N} \frac{y_j}{W(t_j)}[W'(t_i)\delta_{ij} + W(t_i)D_{ij}] = \sum_{j=0}^{N} D_{ij}[W]y_j \tag{14}$$

where we use $D_{ij}[W]$ as a shorthand notation for the W-weighted differentiation matrix,

$$D_{ij}[W] := \frac{[W'(t_i)\delta_{ij} + W(t_i)D_{ij}]}{W(t_j)} \tag{15}$$

and D_{ij} is usual unweighted differentiation matrix given by,

$$D_{ij} := \left.\frac{d\phi_j(t)}{dt}\right|_{t=t_i} \tag{16}$$

Thus, when $W(t) = 1$, we have

$$D_{ij}[1] = D_{ij} \tag{17}$$

From Eq. (13), the unweighted differentiation matrix, $D_{ij} = \phi_j'(t_i)$, has the form,

$$D_{ij} = \begin{cases} \dfrac{g_N'(t_i)}{g_N'(t_j)} \dfrac{1}{(t_i - t_j)}, & i \neq j \\ \\ \dfrac{g_N''(t_i)}{2g_N'(t_i)}, & i = j \end{cases} \tag{18}$$

These equations are the general representations of the derivative of the Lagrange polynomials evaluated at arbitrary interpolation nodes. Thanks to Runge, it is well-known[24] that an improper selection of the grid points can lead to disastrous consequences. In fact, a uniform distribution of grid points is the worst possible choice for polynomial interpolation and hence differentiation. On the other hand, the best possible choice of grid points for integration, differentiation and interpolation of functions are Gaussian quadrature points. Consequently, all PS methods use Gaussian quadrature points.

As an example of the preceding ideas, in the Legendre PS method, the grid points are the shifted Legendre-Gauss-Lobatto (LGL) points where the "shift" is achieved by mapping the physical domain, $[t_0, t_f] \ni t$, to a computational domain, $[-1, 1] \ni \tau$, by the affine transformation,

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

where we have abused notation in using τ to imply both the transformation as well as the transformed variable. The LGL weights and the differentiation matrix are,

$$\begin{aligned} w_k &:= \frac{t_f - t_0}{N(N+1)} \frac{1}{[\mathsf{L}_N(\tau_k)]^2} \quad k = 0, 1, \ldots, N \\ D_{kl}^N &:= \frac{2}{t_f - t_0} \begin{cases} \frac{\mathsf{L}_N(\tau_k)}{\mathsf{L}_N(\tau_l)} \cdot \frac{1}{\tau_k - \tau_l} & k \neq l \\ -\frac{N(N+1)}{4} & k = l = 0 \\ \frac{N(N+1)}{4} & k = l = N \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where $\tau_k, k = 0, 1, \ldots, N$ denote the LGL nodes[23] and $L_N(t)$ denotes the Legendre polynomial of order N.

In the following sections we will denote by $[y_k]$ the collection of the discretized continuous-time variable for $k = 0, \ldots, N$.

B. Pseudospectral Relaxation

Let

$$\boldsymbol{x}^N(t) := \sum_{l=0}^{N} \boldsymbol{x}_l \phi_l(t),\ \ \boldsymbol{u}^N(t) := \sum_{l=0}^{N} \boldsymbol{u}_l \phi_l(t), \tag{19}$$

where $\boldsymbol{x}_l,\ \boldsymbol{u}_l,\ l = 0,\dots,N$ solve the "relaxed" discretized problem,

$$\mathsf{B}^N(\varepsilon^N)\begin{cases} \textsf{Minimize} & J^N([\boldsymbol{x}_k],[\boldsymbol{u}_k],t_0,t_f) = \\ & E(\boldsymbol{x}_0,\boldsymbol{x}_N,t_0,t_f) + \displaystyle\sum_{l=0}^{N} F(\boldsymbol{x}_l,\boldsymbol{u}_l) w_l \\ \textsf{Subject to} & \mathbf{f}(\boldsymbol{x}_k,\boldsymbol{u}_k) - \displaystyle\sum_{l=0}^{N} D_{kl}\boldsymbol{x}_l \in \boldsymbol{\epsilon}_{N_x} \\ & \mathbf{e}\big(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f\big) \in \boldsymbol{\epsilon}_{N_e} \\ & \mathbf{h}(\boldsymbol{x}_k,\boldsymbol{u}_k) \in (-\infty, \boldsymbol{\epsilon}_{N_h}] \\ & \qquad k = 0,1,\dots,N \end{cases}$$

where $\boldsymbol{\epsilon}_{N_z},\ z = x,e,h$ are epsilon sets,

$$\boldsymbol{\epsilon}_{N_z} = \left\{ \boldsymbol{\epsilon} \in \mathbb{R}^{N_z}_{++} : \|\boldsymbol{\epsilon}\|_\infty \le \varepsilon^N \right\} \tag{20}$$

where $\varepsilon^N > 0$ is the relaxation parameter whose implications will be apparent shortly. Strictly speaking, we must write $\boldsymbol{\epsilon}_{N_z}$ as $\boldsymbol{\epsilon}_{N_z}(\varepsilon^N)$ to indicate that it is a set-valued map; however, for the purpose of notational simplicity, we avoid this elaboration.

As noted in Section I and elsewhere,[25] it is critical to guarantee the existence of a solution to the discretized problem if a solution to the original problem (Problem B) exists. The early theory of PS methods for optimal control was based on setting $\varepsilon^N = 0$. Since it is impossible to set $\varepsilon^N = 0$ on a digital computer, the prevailing wisdom was to set it equal to as small a number as possible. The counter example in Ref. [25] shows that by setting ε^N too small may actually result in an infeasible set for the discretized problem. Similar counter examples have been developed by Mordukhovich[18] for Euler (and hence, Runge-Kutta) methods. Setting ε^N too large to "fix the problem" will render the discrete solution infeasible with respect to the original continuous-time problem. Such results have shown that the theory for PS optimal control cannot be directly "lifted" from its corresponding theory employed to solve fluid mechanics problems.[23] In other words, it was necessary to develop a new theory for PS methods to handle the various nuances of optimal control theory.

C. Existence and Convergence

Recent results on PS optimal control[25–27] demonstrate a strong relation between the number of nodes and the relaxation parameter ε^N. The result, posited in the form of the following lemma, not only closes the gap between theory and practice, it also clarifies a number apparent anomalies in the practice of PS discretizations.

Lemma 1 [26]*[Existence] Given any feasible solution, $t \mapsto (x,u)$, for Problem B, suppose $x(\cdot) \in W^{m,\infty}$ with $m \ge 2$. Then, there exists a positive integer N_ε such that, for any $N > N_\varepsilon$, the feasible set of relaxed discretized problem (Problem $B^N(\varepsilon^N)$) is nonempty.*

The Lemma theoretically guarantees the well-posedness of PS optimal control methods. It is also a key result that will be used later to construct the spectral algorithm. Although Lemma 1 is revealing, it does not yet complete the practical foundation in solving the optimal control problem since we need a connection between a discretized solution, $(\boldsymbol{x}_l,\boldsymbol{u}_l)$, and the typically-unknown optimal solution, $(\boldsymbol{x}^*(t_l),\boldsymbol{u}^*(t_l))$. This connection, obtained in Refs. [25–27] ensures the convergence of $\boldsymbol{x}_l$ to $\boldsymbol{x}^*(t_l)$, in addition to the convergence of the controls as well as the dual variables under the following assumption (for the purposes of brevity, we state the conditions for only the state and control variables):

Assumption 1 It is assumed that the sequence $\{\boldsymbol{x}_0\}_N$ has a limit point as $N \to \infty$. Furthermore, there exist continuous functions $\mathbf{q}(t) \in \mathbb{R}^{N_x}$, $\boldsymbol{u}^\infty(t) \in \mathbb{R}^{N_u}$ such that

$$\lim_{N\to\infty} (\dot{\boldsymbol{x}}^N(t), \boldsymbol{u}^N(t)) \quad = \quad (\mathbf{q}(t), \boldsymbol{u}^\infty(t)) \tag{21}$$

uniformly on $t \in [-1, 1]$, where $(\boldsymbol{x}^N(t), \boldsymbol{u}^N(t))$ are defined in (19).

Theorem 1 [26]*[Convergence] Under A1, there exists an optimal solution, $(\boldsymbol{x}^*(\cdot), \boldsymbol{u}^*(\cdot))$, to Problem B such that the following limits converge uniformly for $0 \le k \le N$.*

$$\begin{aligned} \lim_{N\to\infty} (\boldsymbol{x}_k^* - \boldsymbol{x}^*(t_k)) &= 0 \\ \lim_{N\to\infty} (\boldsymbol{u}_k^* - \boldsymbol{u}^*(t_k)) &= 0 \end{aligned}$$

From this theorem, it is apparent that if the the relaxation parameter ε^N is chosen to be some fixed constant independent of N, then convergence to the optimal solution is not guaranteed. In other words, if ε^N is too large, we do not have convergence, and if ε^N is too small for a given N, we may have an infeasible discretized space (by Lemma 1). Thus, from an algorithmic perspective, we must choose the relaxation parameter ε^N to vary inversely as some power of N. The exact nature of the choice of this variation will be discussed shortly but it is apparent that our proposed scheme is a spectral analog of the notion of *consistent approximations* discussed in Refs. [17,18,28].

IV. Integrating the Concepts

For a guess-free method to work properly, the continuous-time optimal control problem (Problem B) must have a solution. From Lemma 1, we note that the existence of a solution to Problem B does not automatically imply the existence of a solution to Problem $B^N(\varepsilon^N)$ for an arbitrary choice of ε^N and N; hence, we need to find consistent pairs, (ε^N, N). By Lemma 1, we also know that for any given ε^N, we can always find an N such that the feasible set of Problem $B^N(\varepsilon^N)$ is nonempty. Alternatively, we can find a ε^N for any given N. Because we choose to solve Problem B via a spectral algorithm, we prefer to select N_0 of the spectral algorithm and find a consistent ε_0. We do this by an "elastic procedure" defined below.

A. Elastic Programming for Problem B^{N_0}

In the first step, we arbitrarily select an initial grid size, N_0, and a finite sequence of increasing numbers, $\varepsilon^0 < \varepsilon^1 < \varepsilon^2 \ldots$, where ε_0 is a desired feasibility tolerance. Such algorithmic tuning parameters may be user-specified. The initialization procedure is to to find a small $\varepsilon^{N_0} > 0$ such that Problem $B^{N_0}(\varepsilon^{N_0})$ is feasible. In our elastic procedure, we start the initialization loop by solving Problem $B(\varepsilon^{N_0})$ for $\varepsilon^{N_0} = \varepsilon^0$. If a feasible or an optimal solution is obtained, we proceed to the next step of the main algorithm; otherwise, a search for a feasible solution for Problem $B(\varepsilon^{N_0})$ is performed by solving the sequence of problems generated by $\varepsilon^{N_0} = \varepsilon^1, \varepsilon^2 \ldots$ until a feasible solution is obtained or the sequence ends due to its finiteness. In the latter case, the optimal control problem is declared infeasible and the algorithm terminates. If the initialization procedure is successful, we denote the corresponding $\varepsilon^{N_0} = \varepsilon^i$ and the spectral algorithm[29] is initiated.

B. Robustness via the Covector Mapping Principle (CMP)

According to the CMP (see Ref. [20] for an introduction to this concept and Ref. [19] for a historical account), a discretization must satisfy dual consistency in order to be convergent for optimal control. This principle along with classic consistency[26,28] provide two key links for overcoming the curse of complexity.

As illustrated in Fig. 3, the elastic program is initiated for solving Problem B^N. Let $\mathbb{M}^{N\lambda}(\chi)$ denote the multiplier set for Problem B^N for any given value of primals, χ; then, it can be shown[12,26] that

$$\mathbb{M}^{N\lambda}(\chi) \supseteq \mathbb{M}^{\lambda N}(\chi)$$

where $\mathbb{M}^{\lambda N}(\chi)$ is the multiplier set associated with solving the boundary value problem (see Figs. 1 and 2). The excess of multipliers in $\mathbb{M}^{N\lambda}(\chi)$ facilitate a natural relaxation to the unstable poles associated with the symplectic structure of the Hamiltonian system represented by Problem B^λ. This relaxation is in addition

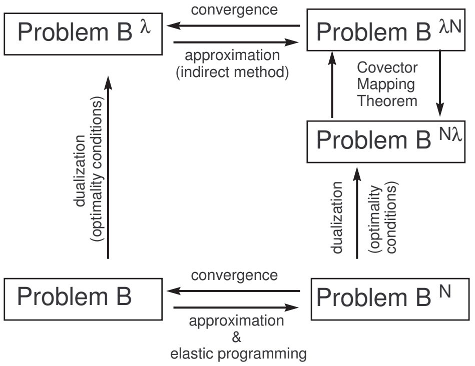

Figure 3. Avoiding the curse of complexity via the CMP; adapted from Ross and Fahroo.[5,12,19]

to PS relaxation discussed in Sec. III.B. In other words, the excess of robustness afforded in both the primal and dual spaces provides a potent way to circumvent the curse of complexity. See Refs. [20] and [26] for more details.

C. Initializing Problem B^{N_0}

It is now clear that if the initialization of Problem $B^{N_0}(\varepsilon^j)$ is done consistent with the spectral algorithm, the algorithmic guarantees provided in the previous sections produce a globally convergent trajectory optimization algorithm. To guarantee a reasonably successful initialization, we assume that the optimal solution to the continuous-time problem (Problem B) is is bounded within sufficiently large state and control spaces; that is, we assume that the optimal solution, $t \mapsto (\boldsymbol{x}, \boldsymbol{u})$ is in $(\mathbb{X} \times \mathbb{U}) \subseteq (\mathbb{R}^{N_x} \times \mathbb{R}^{N_u})$, where $\mathbb{X}$ and $\mathbb{U}$ are bounded. For the purpose of simplicity we take $\mathbb{X}$ and $\mathbb{U}$ to be specified in terms of the box constraints,

$$\mathbb{X} = \left\{ \boldsymbol{x} : \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U \right\} \tag{22}$$

$$\mathbb{U} = \left\{ \boldsymbol{u} : \boldsymbol{u}^L \leq \boldsymbol{u} \leq \boldsymbol{u}^U \right\} \tag{23}$$

In principle, with the addition of this reasonable-bound assumption, the spectral algorithm is now capable of providing solutions from arbitrary points that satisfy,

$$\boldsymbol{x}_k \in \mathbb{X}, \quad \boldsymbol{u}_k \in \mathbb{U}, \quad k = 0, \ldots, N_j$$

We illustrate this feature by solving a problem from robotics.[30] The system is a one-link flexible robot arm whose dynamics are,

$$I_1 \ddot{q}_1 + m_1 g l \sin q_1 + k(q_1 - q_2) = 0 \tag{24}$$

$$I_2 \ddot{q}_2 - k(q_1 - q_2) = u \tag{25}$$

where

$$I_1 = I_2 = 1.0,\ k = 1.0,\ g = 9.8,\ m_1 = 0.01,\ l = 0.5$$

The optimal control problem is to minimize,

$$\int_{\tau_0}^{\tau_f} u^2(\tau)\, d\tau \tag{26}$$

subject to the dynamics, the endpoint constraints,

$$[q_1(\tau_0), q_2(\tau_0), \dot{q}_1(\tau_0), \dot{q}_2(\tau_0)] = [0.03, 0.01, 0.04, 0.05] \tag{27}$$

$$[q_1(\tau_f), q_2(\tau_f), \dot{q}_1(\tau_f), \dot{q}_2(\tau_f)] = [0.06, 0.02, 0.08, 0.02] \tag{28}$$

and the control constraint,

$$\mathbb{U} = \{ u : \ -15 \leq u \leq 15 \} \tag{29}$$

Because this problem already has a box constraint on the controls, we only need to provide a reasonable bound for the states. With a modicum of analysis, it is not too hard to conclude that this reasonable bound may be stipulated as,

$$-20 \le x_i \le 20 \quad i = 1, \ldots, 4$$

Thus, the "search space" for this problem is now shrunk from a possibly four dimensional infinite box to one substantially smaller. Using random numbers inside this box, the PS spectral algorithm converges. That is, for 100 random initial points, we obtained the exact same answer each time without any difficulty. The average run time per run was a mere 0.2 seconds on a 2.4 GHz Pentium 4 PC with 512 MB RAM. See Ref. [10] for further details and analysis.

It is extremely important to note that we are not claiming such performance results for all trajectory optimization problems. We merely discuss this problem to note that the spectral algorithm generates results in a robust manner despite that this problem is considered difficult in robotics. In many applications, it not necessary to start at random points inside $\mathbb{X} \times \mathbb{U}$; it is quite sufficient to start at some arbitrary point that works. Such a working point over the space of all optimal control problems is the center of the box. This is because at worst, the optimal solution points are at the corners of the box which are equidistant from the center. This equidistant notion holds most effectively if the box is reasonably square. This implies that the problem must be reasonably scaled. Since a trajectory optimization problem may indeed be badly scaled in standard units (such as metric units), it is possible to re-scale the problem in "designer units" to meet this assumption for guess-free trajectory optimization.

D. Squaring the Box Using Designer Units

Consider the following equations,

$$\dot{x} = v_x \tag{30}$$

$$\dot{y} = v_y \tag{31}$$

Suppose that the x-motion is very large compared to the y-motion. For instance, x may be in several kilometers while y may be in several meters or fractions of a meter. Then, no matter what distance unit is used, the problem will be badly scaled. While many physicists and engineers prefer to use canonical or similar units to identify a minimal set of parameters for a problem, such units may not necessarily be suitable from a computational point of view because the problem may remain badly scaled even in canonical units. From a computational and mathematical perspective, there is no reason why the distance units for x must be the same as the distance units for y. Thus, for example, we may compute x in kilometers and y in meters. In the same spirit there is no need to have velocity units in the x direction to be the same as that in the y-direction or for that matter for velocity units to be equal to distance units per time units! Thus, we can define new variables as,

$$\overline{x} = x/X, \quad \overline{y} = y/Y, \quad \overline{v}_x = v_x/V_x, \quad \overline{v}_y = v_y/V_y$$

where X, Y, V_x and V_y are arbitrary numbers or **designer units**. Similarly let

$$\overline{t} = t/T$$

where T is a designer unit of time; then, denoting $d\overline{x}/d\overline{t}$ as $\dot{\overline{x}}$ etc., we can write,

$$\dot{\overline{x}} = \frac{TV_x}{X}\overline{v}_x \tag{32}$$

$$\dot{\overline{y}} = \frac{TV_y}{Y}\overline{v}_y \tag{33}$$

Note that no substantial gains in scaling the equations may be possible if we choose to define velocity units in terms of distance units per time units. Thus, by appropriately choosing the five quantities, X, Y, V_x, V_y and T *independently*, it is possible to scale the two equations to computationally attractive choices even when the physical problem is badly scaled.

E. Illustrating the Concepts

Consider the following formulation of Bernoulli's Brachistochrone problem,

$$\boldsymbol{x}^T = [x, y, v] \quad \boldsymbol{u} = [\theta]$$

$$(Brac:1)\left\{\begin{array}{lrl} \text{Minimize} & J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), t_f] = & t_f \\ \text{Subject to} & \dot{x} = & v \sin\theta \\ & \dot{y} = & v \cos\theta \\ & \dot{v} = & g \cos\theta \\ & (x_0, y_0, v_0) = & (0, 0, 0) \\ & (x_f, y_f) = & (x^f, y^f) \\ & t_0 = & 0 \end{array}\right.$$

where g is a constant, equal to 9.8 m/s^2 for Earth. It is apparent by inspection that if x^f and y^f are 1 or 10 m, then the problem is well-scaled in metric units. To apply the guess-free algorithm, we now need to define appropriate box constraints,

$$\begin{array}{rcl} \mathbb{X} & = & \left\{\boldsymbol{x} : \boldsymbol{x}^L \le \boldsymbol{x} \le \boldsymbol{x}^U\right\} \\ \mathbb{U} & = & \left\{\theta : \theta^L \le \theta \le \theta^U\right\} \\ t_f & \in & \left\{t_f : t_f^L \le t_f \le t_f^U\right\} \end{array}$$

that must be non-binding. Let x^f and y^f be 10m, then, it is apparent that a reasonably "large" non-binding box may be defined as,

$$\boldsymbol{x}^L = \boldsymbol{0}, \quad \boldsymbol{x}^U = \boldsymbol{20}, \quad \theta^L = 0, \quad \theta^U = \pi, \quad t_f^L = 0, \quad t_f^U = 10$$

With these numbers, the guess-free spectral algorithm runs successfully and the results are shown in Fig. 4. Also shown in Fig. 5 are the costates and the Hamiltonian. The constancy of the Hamiltonian at the value -1 is an indicator of the extremality of the computed solution while the values of the costates indicates that the dual variables are also well scaled. That is, the problem with the stipulated numbers is also well balanced. This is why we call this the "good" Brachistochrone problem.[31]

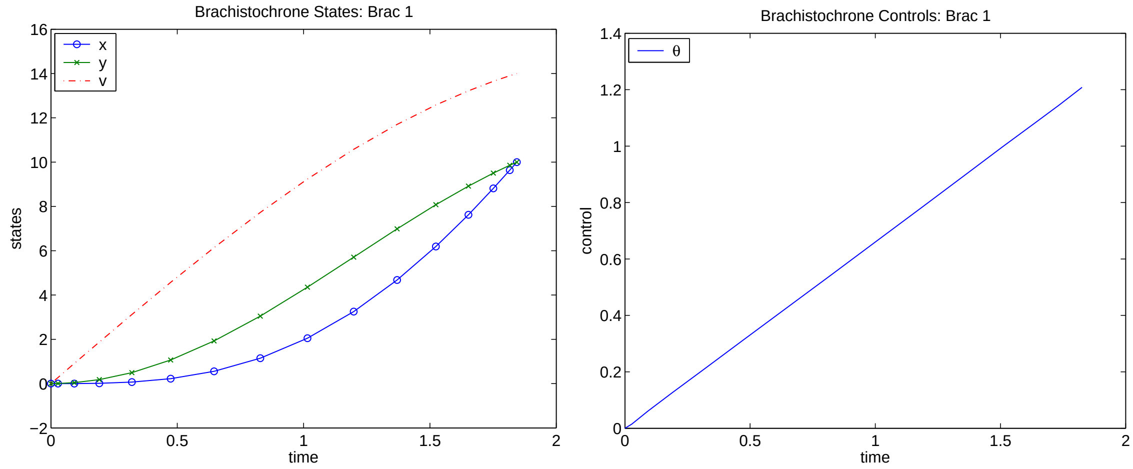

Figure 4. State and control trajectory from a guess-free algorithm implemented in DIDO©.

Now suppose that

$$x^f = 1\ km = 1000m \quad \text{and} \quad y^f = 1m$$

This problem with these numbers is very badly scaled in that we seek to find the minimum time trajectory where the downrange is 1 km (or 1000 m) while the altitude drop is only 1 m. Theoretically, these wild numbers makes no difference, but computationally, the problem becomes difficult to manage; hence, we call this the "bad" Brachistochrone problem.[31]

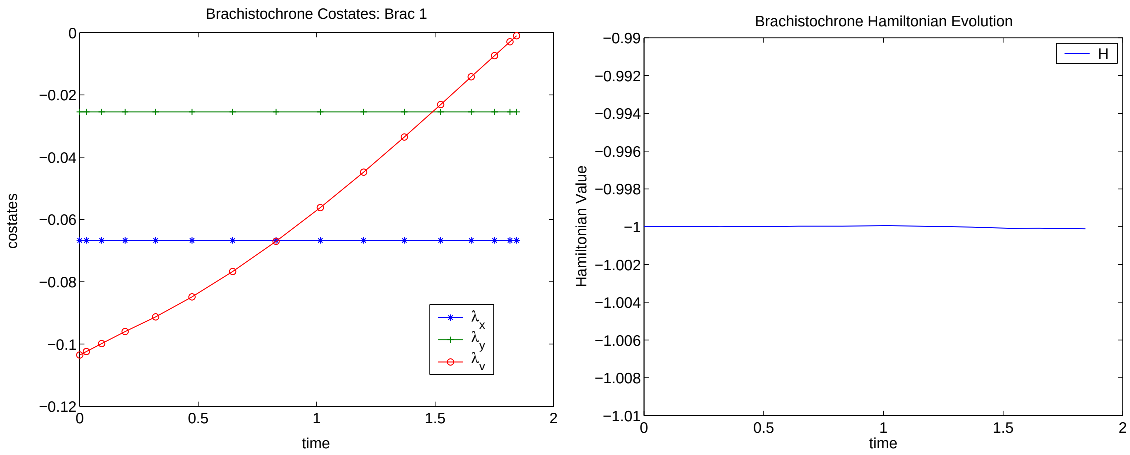

Figure 5. Costate trajectory and Hamiltonian evolution from a guess-free algorithm implemented in DIDO.

Let

$$\overline{\boldsymbol{x}}^T = [x/X, y/Y, v/V] \quad \overline{\boldsymbol{u}} = [\theta/\Theta] \quad \overline{t} = t/T$$

where X, Y, V, Θ and T are arbitrary numbers or **designer units**. Denoting $d\overline{x}/d\overline{t}$ as $\dot{\overline{x}}$ etc., we have a potentially well-scaled problem given by,

$$(\overline{Brac}:1)\begin{cases} \textsf{Minimize} \quad \overline{J}[\overline{\boldsymbol{x}}(\cdot), \overline{\boldsymbol{u}}(\cdot), \overline{t}_f] & = \overline{t}_f \\ \textsf{Subject to} \qquad\qquad \dot{\overline{x}} & = \dfrac{TV}{X}\overline{v}\sin(\overline{\theta}\Theta) \\ \dot{\overline{y}} & = \dfrac{TV}{Y}\overline{v}\cos(\overline{\theta}\Theta) \\ \dot{\overline{v}} & = \dfrac{Tg}{V}\cos(\overline{\theta}\Theta) \\ (\overline{x}_0, \overline{y}_0, \overline{v}_0) & = (0,0,0) \\ (\overline{x}_f, \overline{y}_f) & = \left(\dfrac{x^f}{X}, \dfrac{y^f}{Y}\right) \\ \overline{t}_0 & = 0 \\ \overline{t}_f & \le \dfrac{t^U}{T} \end{cases}$$

Note that the cost function in this formulation $(\overline{J})$ is scaled by T. As will be apparent shortly, this implies that the covectors also get scaled. Furthermore, the cost function may also be scaled by any value (and not necessarily, T) because minimizing a function or some multiple of the function generates the same value for the independent variable (but not necessarily the cost value). In any event, the problem in its metric units is easily recovered by setting all values of the designer units to be one. Now, one might hastily (and falsely) conclude that we wish to choose $X = 1000$ and $Y = 1$ so that $\overline{x}$ and $\overline{y}$ range from 0 to 1. *One of the most important reasons why this is false is that the problem units must be balanced in dual space as well.*

F. Dual Variables and Units: They Do Have a Meaning!

The control Hamiltonian for the scaled problem is given by,

$$H(\overline{\boldsymbol{\lambda}}, \overline{\boldsymbol{x}}, \overline{\boldsymbol{u}}) := \overline{\lambda}_x \frac{TV}{X}\overline{v}\sin(\overline{\theta}\Theta) + \overline{\lambda}_y \frac{TV}{Y}\overline{v}\cos(\overline{\theta}\Theta) + \overline{\lambda}_v \frac{Tg}{V}\cos(\overline{\theta}\Theta)$$

From the Hamiltonian value condition and the Hamiltonian evolution equation, we have

$$H(\overline{\boldsymbol{\lambda}}(t), \overline{\boldsymbol{x}}(t), \overline{\boldsymbol{u}}(t)) = -1$$

This implies,

$$\overline{\lambda}_x \frac{TV}{X} \overline{v} \sin(\overline{\theta}\Theta) + \overline{\lambda}_y \frac{TV}{Y} \overline{v} \cos(\overline{\theta}\Theta) + \overline{\lambda}_v \frac{Tg}{V} \cos(\overline{\theta}\Theta) = -1 \tag{34}$$

Now, in the "good" Brachistochrone problem, we had,

$$x^f = 10\ m \quad \text{and} \quad y^f = 10\ m$$

and all units were "1". An inspection of Figure 4 shows that v and hence, $\overline{v}$, varies from 0 to about 15. From Equation 34, this implies that we should expect the costates to vary from 0 to about 0.1. This is precisely the result in Figure 5. It is clear that the metric units would generate a badly scaled problem for the bad Brachistochrone problem. A fundamental rule for balancing equations is to choose designer units in such a manner that the states and costates are roughly the same order of magnitude. As noted by Ross,[31] the covectors do have physical meaning with **co-units** given by,

$$\boldsymbol{\lambda}\ \text{UNITS} = \frac{\text{Cost UNITS}}{\text{State UNITS}} \tag{35}$$

Consequently, the units for the adjoint covectors are automatically set when one chooses the units for the states; that is, we have,

$$\overline{\lambda}_x = \lambda_x \frac{X}{T} \qquad \overline{\lambda}_y = \lambda_y \frac{Y}{T} \qquad \overline{\lambda}_v = \lambda_v \frac{V}{T} \tag{36}$$

As an example, if we choose

$$X = 10 = Y = V$$

in the good Brachistochrone problem with all other units as before (i.e. 1, or metric), then the state variables will vary from approximately 0 to 1 (because of the "physics" of the problem and its data; see Figure 4). Then, according to Equation 36, the costates must vary from approximately 0 to 1 as well (Cf. Figure 5). This analysis is borne out in Figure 6. Because the good Brachistochrone problem is already well scaled in metric units the guess-free algorithm works just fine in either units. Note also that the designer units used above are not the usual "canonical" units typically used in the literature. If we were to choose canonical units, we must set,

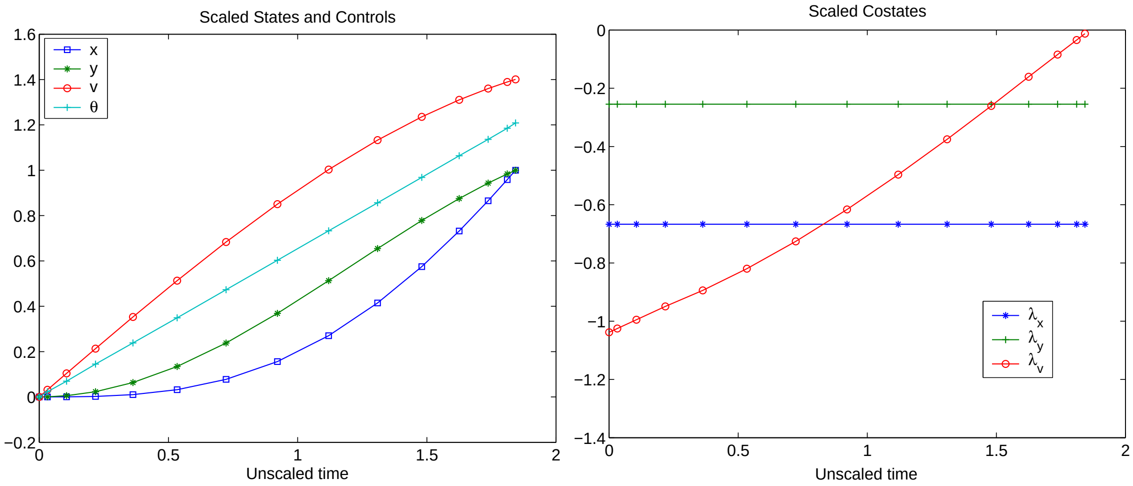

Figure 6. Scaled state and costate trajectory from a guess-free algorithm implemented in DIDO; compare Figures 4 and 5.

$$X = Y, \quad V = \frac{X}{T}, \quad T = \frac{V}{g} = \frac{X}{gT}$$

In this case, all units get set automatically if any one of them is chosen to be a specific number. For example, choosing $X = 1m$ we get,

$$X = Y = 1\ m, \quad T = \sqrt{\frac{1}{9.81}} \approx 0.32s, \quad V = \frac{X}{T} \approx 3.1m/s$$

which has the effect of expanding the time scale by about a third, and hence shrinking the velocity scale by about a third. A better choice of canonical units might be to choose $X = x^f = 10\,m$; in this case, the time unit is nearly 1 and the velocity unit is nearly 10 and all the variables range from approximately 0 to 1 and the results will be nearly the same as that illustrated in Figure 6.

It is clear that neither the metric units nor canonical units would generate a well-scaled problem for the bad Brachistochrone problem. Furthermore, if were to choose designer units $X = 1000$ and $Y = 1$ so that $\overline{x}$ and $\overline{y}$ range from 0 to 1, the variables are well scaled but the problem is badly balanced as this implies that $\overline{\lambda}_x$ would be multiplied by 1000 (everything else remaining the same). Thus, a more informative choice for the designer units might be

$$X = 100, \quad Y = 20, \quad V = 10, \quad \Theta = 1, \quad T = 10$$

A solution to the bad Brachistochrone problem in these designer units is shown in Fig. 7. Note that the costate λ_v in designer units is indeed exactly equal to its value in metric units. This follows from Equation 36 and our choice of $V = T = 10$. Furthermore because $X = 100$, the value of λ_x in designer units (= -0.127) is ten times larger (in absolute value) than it metric value of -0.0127 s/m. Likewise, because $Y = 20$, the value of λ_y (= 0.4510) in designer units is two times larger (in absolute value) than its metric value of 0.2255 s/m.

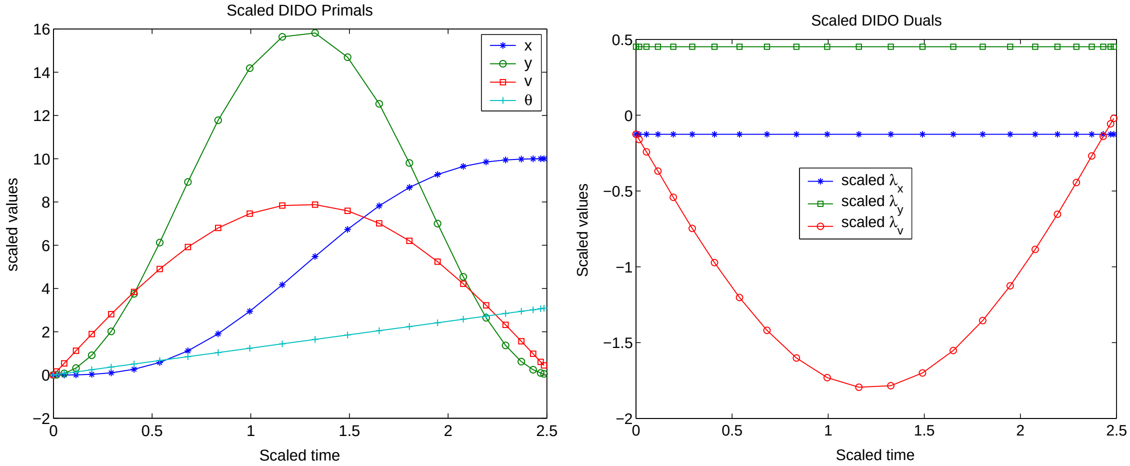

Figure 7. Scaled and balanced variables in designer units solved by DIDO's guess-free algorithm.

This solution was obtained by setting the non-binding box constraints as,

$$\overline{\boldsymbol{x}}^L = \mathbf{0}, \quad \overline{\boldsymbol{x}}^U = \mathbf{200}, \quad \overline{\theta}^L = 0, \quad \overline{\theta}^U = \pi, \quad \overline{t}_f^L = 0, \quad \overline{t}_f^U = 20$$

The solution in metric units is shown in Figure 8. It is worth observing that the guess-free algorithm also works even when the non-binding box constraints are non-square and substantially enlarged to

$$\overline{\boldsymbol{x}}^L = \mathbf{0}, \quad \overline{\boldsymbol{x}}^U = (1000, 200, 1000), \quad \overline{\theta}^L = 0, \quad \overline{\theta}^U = \pi, \quad \overline{t}_f^L = 0, \quad \overline{t}_f^U = 20$$

V. Additional Examples

The guess-free algorithm is now widely used[a] by way of its implementation in DIDO. That is, DIDO runs without the aid of a guess, while not preventing a user by supplying one. As a result, some interesting

[a] See http://www.elissar.biz.

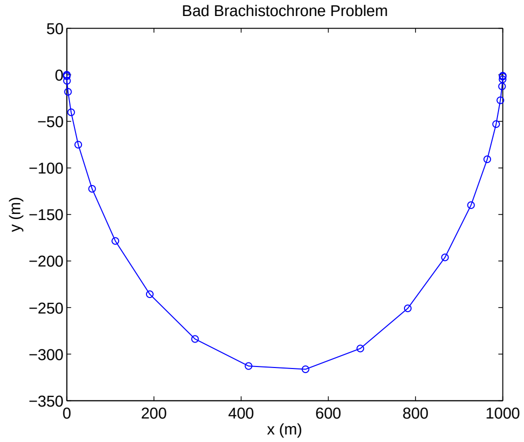

Figure 8. Solution to the Bad Brachistochrone Problem from DIDO's guess-free algorithm.

comments have been reported by various users around the world. For instance, in many situations, users have reported that DIDO runs faster in its guess-free mode than when supplied with one. In any event, fuel-optimal solutions to a 4-body problem under the guess-free algorithm are discussed and provided by Park et al.[32] Hurni et al[33] discuss how to use the guess-free algorithm to autonomously plan and execute the motion of an unmanned ground vehicle, particularly when information about obstacles is unknown *a priori*. Finally, additional examples are provided are provided as part of the DIDO software package.[31]

VI. Conclusions

The twin curses of complexity and dimensionality have long hampered a viable approach to a guess-free trajectory optimization algorithm. Advances in pseudospectral (PS) methods indicate that PS discretization offer a certain robustness to the solution of the discrete problem by way of the covector mapping principle. These advances are made possible by a combination of ideas: PS relaxation related to the existence and convergence of the discretization, the use of the Covector Mapping Principle (CMP) for designing the spectral algorithm, and elastic programming for an initial setup. When these advances are combined together, the design of a guess-free algorithm relies on a mild assumption that the solution is reasonably bounded and that the problem variables are well scaled and balanced. For any given problem, it is possible to computationally balance the vectors and covectors using Pontryagin's Principle as a necessary condition rather than as a problem solving concept. The CMP does the remainder of the tasks.

References

[1] Ortega, J. M., "The Newton-Kantorovich Theorem," *The American Mathematical Monthly*, Vol. 75, No. 6, Jun. - Jul., 1968, pp. 658-660.

[2] Gragg, W. B., and Tapia, R. A., "Optimal Error Bounds for the Newton-Kantorovich Theorem," *SIAM Journal on Numerical Analysis*, Vol. 11, No. 1, 1974, pp. 10-13

[3] Dennis, J. E. and Schnabel, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[4] Kelley, C. T., *Solving Nonlinear Equations with Newtons Method*, SIAM, Philadelphia, 2003.

[5] Ross, I. M., and Fahroo, F., "A Perspective on Methods for Trajectory Optimization," *Proceedings of the AIAA/AAS Astrodynamics Conference*, Monterey, CA, August 2002. Invited Paper No. AIAA 2002-4727.

[6] Fahroo, F. and Ross, I. M., "Advances in Pseudospectral Methods for Optimal Control," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 18-21, 2008. AIAA Paper 2008-7309.

[7] Kang, W. and Bedrossian, N., "Pseudospectral Optimal Control Theory Makes Debut Flight," *SIAM News, Volume 40, Number 7, September 2007*, pg. 1.

[8] Bedrossian, N., Bhatt, S., Lammers M., Nguyen, L., Zhang, Y., "First Ever Flight Demonstration of Zero Propellant Maneuver Attitude Control Concept," AIAA-2007-6734.

[9] Bedrossian, N., Bhatt, S., Lammers M., Nguyen, L., "Zero Propellant Maneuver: Flight Results for 180° ISS Rotation," *20th International Symposium on Space Flight Dynamics*, September 24-28, 2007, Annapolis, MD, NASA/CP-2007-214158.

[10] Ross, I. M. and Fahroo, F., "Issues in the Real-Time Computation of Optimal Control," *Mathematical and Computer Modelling*, An International Journal, Vol. 43, Issues 9-10, May 2006, pp.1172-1188. (Special Issue: Optimization and Control

for Military Applications).

[11] Hager, W. W., "Numerical Analysis in Optimal Control," *International Series of Numererical Mathematics*, Hoffmann, K.-H. Lasiecka, I., Leugering, G., Sprekels, J., and Troeltzsch, F., Eds., Birkhäuser, Basel, Switzerland, 2001, Vol. 139, pp. 83–93.

[12] Ross, I. M., and Fahroo, F., "Legendre Pseudospectral Approximations of Optimal Control Problems," *Lecture Notes in Control and Information Sciences*, Vol. 295, Springer-Verlag, New York, 2003, pp. 327–342.

[13] Betts, J. T., *Practical Methods for Optimal Control Using Nonlinear Programming*, SIAM, Philadelphia, PA, 2001.

[14] Ross, I. M., "Addressing Dimensionality and Complexity Curses in Computational Optimal Control," *SIAM Conference on Computational Science and Engineering*, Feb. 19-23, 2007, Costa Mesa, CA.

[15] McEneaney, W. M., "A Curse-of-Dimensionality-Free Numerical Method for Solution of Certain HJB PDEs," *SIAM Journal of Control and Optimization*, Vol. 46, Issue 4, 2007, pp. 1239-1276.

[16] Hairer E., Lubich C, and Wanner G., *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer, 2006.

[17] Mordukhovich, B. S., *Variational Analysis and Generalized Differentiation, I: Basic Theory*, vol. 330 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences] Series, Springer, Berlin, 2005.

[18] Mordukhovich, B. S., *Variational Analysis and Generalized Differentiation, II: Applications*, vol. 331 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences] Series, Springer, Berlin, 2005.

[19] Ross, I. M.,"A Historical Introduction to the Covector Mapping Principle," *Advances in the Astronautical Sciences*, Vol. 123, Univelt, San Diego, CA, 2006, pp. 1257-1278.

[20] Ross, I. M., "A Roadmap for Optimal Control: The Right Way to Commute," *Annals of the New York Academy of Sciences*, Vol. 1065, New York, N.Y., December 2005.

[21] Fahroo, F. and Ross, I. M., "On Discrete-Time Optimality Conditions for Pseudospectral Methods," *Proceedings of the AIAA/AAS Astrodynamics Conference*, Keystone, CO, August 2006. AIAA-2006-6304

[22] Fahroo, F. and Ross, I. M., "Pseudospectral Methods for Infinite Horizon Optimal Control Problems," *Journal of Guidance, Control and Dynamics*, July-August, 2008.

[23] Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics.* Springer Verlag, New York, 1988.

[24] Trefethen, L. N., *Spectral Methods in MATLAB,* SIAM, Philadelphia, PA, 2000.

[25] Gong, Q., Kang, W., and Ross, I. M., A pseudospectral method for the optimal control of constrained feedback linearizable systems, *IEEE Trans. Auto. Cont.*, Vol. 51, No. 7, July 2006, pp. 1115-1129.

[26] Gong, Q., Ross, I. M., Kang, W., and Fahroo, F., Connections between the covector mapping theorem and convergence of pseudospectral methods for optimal control, *to appear in Computational Optimization and Applications*, 2008.

[27] Kang, W., Gong, Q., Ross, I. M., and Fahroo, F., On the convergence of nonlinear optimal control using pseudospectral methods for feedback linearizable control systems, *International Journal on Robust and Nonlinear Control*, Vol. 17, Issue 14, pp. 1251-1277, 2007.

[28] Polak, E., *Optimization: Algorithms and Consistent Approximations*, Springer-Verlag, Heidelberg, 1997.

[29] Gong, Q., Fahroo, F. and Ross, I. M., A spectral algorithm for pseudospectral methods in optimal control, *AIAA Journal of Guidance, Control and Dynamics,* Vol. 31, No. 3, pp. 460-471, 2008.

[30] Isidori, A., *Nonlinear Control Systems,* Springer-Verlag, New York, NY, 1989.

[31] Ross, I. M., "A Beginner's Guide to DIDO: A MATLAB application package for solving optimal control problems," Elissar, Monterey, CA, 2007.

[32] Park, C., Gong, Q., Ross, I. M., and Sekhavat, P., "Fuel-Optimal Design of Moon-Earth Trajectories Using A Legendre Pseudospectral Method," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Honolulu, Hawaii, August 18-21, 2008. AIAA Paper 2008-7074.

[33] Hurni, M. A., Ross, I. M., Sekhavat, P., "Autonomous Path Planning Using Real-Time Information Updates," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 18-21, 2008. AIAA Paper 2008-6305.