2018-10

Scaling and Balancing for High-Performance Computation of Optimal Controls

Ross, I.M.; Gong, Q.; Karpenko, M.; Proulx, R.J.

ARC

Ross, I. M., et al. "Scaling and balancing for high-performance computation of optimal controls." Journal of Guidance, Control, and Dynamics 41.10 (2018): 2086-2097. http://hdl.handle.net/10945/66179

JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 41, No. 10, October 2018

Scaling and Balancing for High-Performance Computation of Optimal Controls

I. M. Ross*
Naval Postgraduate School, Monterey, California 93943
Q. Gong†
University of California, Santa Cruz, California 95064
and
M. Karpenko‡ and R. J. Proulx§
Naval Postgraduate School, Monterey, California 93943

DOI: 10.2514/1.G003382

It is well known that proper scaling can increase the efficiency of computational problems. In this paper, we define and show that a balancing technique can substantially improve the computational efficiency of optimal-control algorithms. We also show that noncanonical scaling and balancing procedures may be used quite effectively to reduce the computational difficulty of some hard problems. These results have been used successfully for several flight and field operations at NASA and the U.S. Department of Defense. A surprising aspect of our analysis shows that it may be inadvisable to use autoscaling procedures employed in some software packages. The new results are agnostic to the specifics of the computational method; hence, they can be used to enhance the utility of any existing algorithm or software.

I. Introduction

IN MANY practical optimal-control problems, the decision variables range wildly in several orders of magnitude [1]. For instance, in a space trajectory optimization problem [2,3], a position variable can vary from a few meters to well over a million kilometers. The conventional wisdom to manage the associated computational problems is to use canonical units. Many space trajectory optimizations fare well when canonical units are used [2,3]. For instance, for low-Earth-orbiting spacecraft, one may choose the radius of the Earth $R_\oplus$ as a unit of distance and circular speed (at $R_\oplus$) as the unit of velocity. For interplanetary spacecraft, astronomical units provide a set of canonical units for scaling trajectory optimization problems. In this paper, we show that it is possible to define and use arbitrary and inconsistent units for faster trajectory optimization. For example, we may choose meters as a unit of distance along the x direction while concurrently using feet for distance units along the y direction. Furthermore, we show that it is not necessary to choose a consistent (or canonical) unit of velocity in the x direction to be equal to meters per second (or feet per second for velocity in the y direction). Thus, for example, one may arbitrarily choose yards per day as the unit of the x velocity while insisting that the x position be measured in meters. The purpose of using such unusual or designer units [4], i.e., highly customized units that do not necessarily conform to standardized units, is to liberate ourselves from using well-established canonical/consistent units so that we may scale an optimal-control problem for faster computational results. This liberation allows us to radically alter what we mean by scaling optimal-control problems and consequently solve some apparently hard problems with more ease than ever before.

A second aspect of our paper is the impact of scaling on dual variables. From the Hahn–Banach theorem [5], dual variables exist for all computational optimal-control problems even when they are not used. To illustrate this consequential theorem, consider the ordinary differential equation

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^{N_x}$ and $\boldsymbol{f}: \boldsymbol{x} \mapsto \mathbb{R}^{N_x}$. The formal adjoint to the variation of Eq. (1) is defined by

$$\dot{\boldsymbol{\lambda}} = -[\partial_x \boldsymbol{f}]^T \boldsymbol{\lambda} \tag{2}$$

where $\partial_x \boldsymbol{f}$ is the Jacobian of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$. In other words, Eq. (2) exists from the mere fact that Eq. (1) exists; hence, when a differential equation in a computational optimal-control problem is scaled, it automatically affects the adjoint equation, even if it (i.e., the adjoint equation) is not used. In this paper, we show that the equations in a computational optimal-control problem must be scaled in such a way that they do not unscale the adjoint variable even if the adjoint equation is never used in the algorithm. We call this type of scaling *balancing*.

In most algorithms, including the ones in which derivatives are not used (e.g., genetic algorithms), the information content in the Jacobian forms a key ingredient in the recipe that connects the sequence of iterations [6–8]. Consequently, the scales used in an optimal-control problem must be balanced even if the adjoint equation is never used because it represents the information content contained in a Jacobian by way of Eq. (2). In other words, there is no escape from considering the adjoint equations. This is a fundamental result traceable to the Hahn–Banach theorem. An alternative explanation for the no-escape clause is that the adjoint equations are part of the necessary conditions for optimality. By definition, necessary conditions are indeed necessary; hence, it should not be entirely surprising that balancing is also necessary.

Our analysis also reveals another surprising result: if scaling is performed at the discrete level, it inadvertently introduces new terms in the dynamical equations with possible feedback effects that may destabilize the search algorithm. Consequently, automatic techniques that scale the problem at the discrete level may be more harmful than useful. The simple remedy is to scale and balance the equations at the

Received 17 October 2017; revision received 26 February 2018; accepted for publication 8 March 2018; published online 7 August 2018.

*Distinguished Professor and Program Director, Control and Optimization, Department of Mechanical and Aerospace Engineering.
†Associate Professor, Department of Applied Mathematics.
‡Research Associate Professor and Associate Director, Control and Optimization Laboratories, Department of Mechanical and Aerospace Engineering; mkarpenk@nps.edu (Corresponding Author).
§Research Professor, Control and Optimization Laboratories, Space Systems Academic Group.

optimal-control level and choose algorithms that do not scale the equations at the discrete level. This simple trick has been used many times before by NASA [9–16] and the U.S. Department of Defense [17–22] to solve and implement on-orbit and fielded solutions. Note, however, that scaling and balancing were not at the forefront in many of these applications because their main contributions far outweighed such discussions. For instance, the focal point of [9] was the flight implementation of the optimal propellant maneuver onboard the International Space Station rather than the employment of scaling and balancing techniques. Similarly, [11,12] were focused on the first flight implementations of a historic zero-propellant maneuver while [15] was focused on the feasibility of arcsecond slews for precision pointing of the Kepler spacecraft. In the same spirit, the main contribution of [18] was the flight implementation of a shortest-time maneuver and not on the specifics or the importance of scaling and balancing that were necessary to accomplish the on-orbit demonstration.

In this paper, we generalize the application-specific procedures of the past successes by laying down the mathematical foundations for scaling and balancing of generic optimal-control algorithms. In doing so, we also demonstrate the fallacies of some scaling techniques that are currently in practice.

II. General Problem Formulation

A generic optimal-control problem can be formulated as follows:

$$\left.\begin{array}{ll}\mathbb{X}=\mathbb{R}^{N_x} & \mathbb{U}=\mathbb{R}^{N_u}\\ \boldsymbol{x}=(x_1,\ldots,x_{N_x}) & \boldsymbol{u}=(u_1,\ldots,u_{N_u})\end{array}\right\}\ \text{(preamble)}$$

$$\overbrace{(B)}^{\text{problem}}\left\{\begin{array}{l}\text{Minimize}\\ \left.\begin{array}{l}J[\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),t_0,t_f]:=\\ E(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)+\int_{t_0}^{t_f}F(\boldsymbol{x}(t),\boldsymbol{u}(t),t)\,\mathrm{d}t\end{array}\right\}\ \text{(cost)}\\ \text{Subject to}\\ \left.\begin{array}{l}\dot{\boldsymbol{x}}=\boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t),t)\end{array}\right\}\ \text{(dynamics)}\\ \left.\begin{array}{l}\boldsymbol{e}^L\le\boldsymbol{e}(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)\le\boldsymbol{e}^U\end{array}\right\}\ \text{(events)}\\ \left.\begin{array}{l}\boldsymbol{h}^L\le\boldsymbol{h}(\boldsymbol{x}(t),\boldsymbol{u}(t),t)\le\boldsymbol{h}^U\end{array}\right\}\ \text{(path)}\end{array}\right.$$

Although the symbols used in problem B are fairly standard (see, for example, [4]), we note the following for the purposes of completeness:

1) $\mathbb{X}$ and $\mathbb{U}$ are N_x- and N_u-dimensional real-valued state and control spaces, respectively. We assume $N_x\in\mathbb{N}^+$ and $N_u\in\mathbb{N}^+$.

2) J is the scalar cost function. The arguments of J are the optimization variables.

3) The optimization variables are the N_x-dimensional state trajectory $\boldsymbol{x}(\cdot)$, the N_u-dimensional control trajectory $\boldsymbol{u}(\cdot)$, the initial clock time t_0, and the final clock time t_f.

4) E is the scalar endpoint cost function. The arguments of E are the endpoints. In the classical literature, E is known as the Mayer cost function.

5) The endpoints are the initial state $\boldsymbol{x}_0\equiv\boldsymbol{x}(t_0)$, the final state $\boldsymbol{x}_f\equiv\boldsymbol{x}(t_f)$, the initial time t_0, and the final time t_f.

6) F is the scalar running cost function. The arguments of F are the instantaneous value of the state variable $\boldsymbol{x}(t)$, the instantaneous value of the control variable $\boldsymbol{u}(t)$, and time t.

7) $\boldsymbol{f}$ is the N_x-dimensional dynamics function, or, more appropriately, the right-hand side of the dynamics equation. The arguments of $\boldsymbol{f}$ are exactly the same as the arguments of F.

8) $\boldsymbol{e}$ is the N_e-dimensional endpoint constraint function. The arguments of $\boldsymbol{e}$ are exactly the same as that of E.

9) $\boldsymbol{e}^L$ and $\boldsymbol{e}^U$ are the N_e-dimensional lower and upper bounds on the values of $\boldsymbol{e}$.

10) $\boldsymbol{h}$ is the N_h-dimensional path constraint function. The arguments of $\boldsymbol{h}$ are exactly the same as that of F.

11) $\boldsymbol{h}^L$ and $\boldsymbol{h}^U$ are the N_h-dimensional lower and upper bounds on the values of $\boldsymbol{h}$.

The five functions, E, F, $\boldsymbol{f}$, $\boldsymbol{e}$, and $\boldsymbol{h}$ are collectively known as the data functions (for problem B).

Regardless of any type of method used to solve problem B, including the so-called direct methods, a solution to the problem must at least satisfy the necessary conditions of optimality because they are indeed necessary. The necessary conditions for problem B are generated by an application of Pontryagin's principle [4,23,24]. This results in the following boundary value problem (BVP), which we denote as problem B^λ:

$$(B^\lambda)\left\{\begin{array}{rl}\dot{\boldsymbol{x}}(t)-\partial_{\lambda}\bar{H}(\boldsymbol{\mu}(t),\boldsymbol{\lambda}(t),\boldsymbol{x}(t),\boldsymbol{u}(t),t)=\boldsymbol{0} & \text{(state equations)}\\ \dot{\boldsymbol{\lambda}}(t)+\partial_{x}\bar{H}(\boldsymbol{\mu}(t),\boldsymbol{\lambda}(t),\boldsymbol{x}(t),\boldsymbol{u}(t),t)=\boldsymbol{0} & \text{(costate equations)}\\ \boldsymbol{h}^L\le\boldsymbol{h}(\boldsymbol{x}(t),\boldsymbol{u}(t),t)\le\boldsymbol{h}^U & \text{(path constraint)}\\ \partial_{u}\bar{H}(\boldsymbol{\mu}(t),\boldsymbol{\lambda}(t),\boldsymbol{x}(t),\boldsymbol{u}(t),t)=\boldsymbol{0} & \text{(Hamiltonian}\\ \boldsymbol{\mu}\dagger\boldsymbol{h} & \text{minimization)}\\ \boldsymbol{e}^L\le\boldsymbol{e}(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)\le\boldsymbol{e}^U & \text{(endpoint equations)}\\ \boldsymbol{\lambda}(t_0)+\partial_{x_0}\bar{E}(\boldsymbol{\nu},\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=\boldsymbol{0} & \text{(initial and}\\ \boldsymbol{\lambda}(t_f)-\partial_{x_f}\bar{E}(\boldsymbol{\nu},\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=\boldsymbol{0} & \text{final transversality}\\ \boldsymbol{\nu}\dagger\boldsymbol{e} & \text{conditions)}\\ \mathcal{H}[@t_0]-\partial_{t_0}\bar{E}(\boldsymbol{\nu},\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=0 & \text{(Hamiltonian}\\ \mathcal{H}[@t_f]+\partial_{t_f}\bar{E}(\boldsymbol{\nu},\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=0 & \text{value conditons)}\end{array}\right.$$

In problem B^λ, the unknowns are as follows:

1) The system trajectory is $t\mapsto(\boldsymbol{x},\boldsymbol{u})\in\mathbb{R}^{N_x}\times\mathbb{R}^{N_u}$.
2) The adjoint covector function is $t\mapsto\boldsymbol{\lambda}\in\mathbb{R}^{N_x}$.
3) The path covector function is $t\mapsto\boldsymbol{\mu}\in\mathbb{R}^{N_h}$.
4) The endpoint covector is $\boldsymbol{\nu}\in\mathbb{R}^{N_e}$.
5) The initial and final clock times are $t_0\in\mathbb{R}$ and $t_f\in\mathbb{R}$.

The quantity $\bar{H}$ is the Lagrangian of the Hamiltonian given by

$$\bar{H}(\boldsymbol{\mu},\boldsymbol{\lambda},\boldsymbol{x},\boldsymbol{u},t):=H(\boldsymbol{\lambda},\boldsymbol{x},\boldsymbol{u},t)+\boldsymbol{\mu}^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{u},t)$$

where H is the usual Pontryagin Hamiltonian,

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

and $\mathcal{H}$ is the lower or minimized Hamiltonian,

$$\mathcal{H}(\boldsymbol{\lambda},\boldsymbol{x},t):=\min_{u}H(\boldsymbol{\lambda},\boldsymbol{x},\boldsymbol{u},t)$$

The symbol $\mathcal{H}[@t]$ is a shorthand for $\mathcal{H}(\boldsymbol{\lambda}(t),\boldsymbol{x}(t),t)$. The quantity $\bar{E}$ is the endpoint Lagrangian given by

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

The $\dagger$ notation used in defining problem B^λ denotes complementarity conditions [4]. For instance, $\boldsymbol{\nu}\dagger\boldsymbol{e}$ is a shorthand for the conditions

$$\nu_i\left\{\begin{array}{ll}\le 0 & \text{if } e_i(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=e_i^L\\ =0 & \text{if } e_i^L<e_i(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)<e_i^U\\ \ge 0 & \text{if } e_i(\boldsymbol{x}_0,\boldsymbol{x}_f,t_0,t_f)=e_i^U\\ \text{unrestricted} & \text{if } e_i^L=e_i^U\end{array}\right.\tag{3}$$

with $\boldsymbol{\mu}\dagger\boldsymbol{h}$ defined similarly (for each t).

III. Scaling the Primal Problem

We change the coordinates of the unknown variables of problem B according to the affine transformations,

$$\boldsymbol{x}:=P_x\tilde{\boldsymbol{x}}+\boldsymbol{q}_x\tag{4a}$$

$$\boldsymbol{u} \coloneqq \boldsymbol{P}_u\tilde{\boldsymbol{u}} + \boldsymbol{q}_u \tag{4b}$$

$$t \coloneqq p_t\tilde{t} + q_t \tag{4c}$$

where the uppercase letter $\boldsymbol{P}_{(\cdot)}$ denotes an invertible square matrix of appropriate dimensions and the lower case letter $p_{(\cdot)}$ is a scalar. Similarly, $\boldsymbol{q}_{(\cdot)}$ is a vector of appropriate dimension, and $q_{(\cdot)}$ is a scalar. The tilded ($\sim$) variables are the transformed variables. In a similar fashion, we scale the cost functional J, the endpoint constraint function $\boldsymbol{e}$, and the path constraint function $\boldsymbol{h}$ according to

$$J \coloneqq p_J\tilde{J} + q_J \tag{5a}$$

$$\boldsymbol{e} \coloneqq \boldsymbol{P}_e\tilde{\boldsymbol{e}} + \boldsymbol{q}_e \tag{5b}$$

$$\boldsymbol{h} \coloneqq \boldsymbol{P}_h\tilde{\boldsymbol{h}} + \boldsymbol{q}_h \tag{5c}$$

where $p_J > 0$ and $\boldsymbol{P}_e$ and $\boldsymbol{P}_h$ are positive definite diagonal matrices. Let problem $\tilde{B}$ denote the transformation of problem B resulting from Eqs. (4) and (5). This problem can be explicitly obtained as follows. First, the transformation of J to $\tilde{J}$ can be constructed using Eq. (4) as

$$\begin{aligned} J[\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),t_0,t_f] &= J\Big[\boldsymbol{P}_x\tilde{\boldsymbol{x}}(\cdot)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(\cdot)+\boldsymbol{q}_u,p_t\tilde{t}_0+q_t,p_t\tilde{t}_f+q_t\Big] \\ &\coloneqq p_J\tilde{J}\Big[\tilde{\boldsymbol{x}}(\cdot),\tilde{\boldsymbol{u}}(\cdot),\tilde{t}_0,\tilde{t}_f\Big]+q_J \end{aligned} \tag{6}$$

where the last equality in Eq. (6) follows from Eq. (5). Hence, the cost functional transforms according to

$$\begin{aligned} &\tilde{J}\Big[\tilde{\boldsymbol{x}}(\cdot),\tilde{\boldsymbol{u}}(\cdot),\tilde{t}_0,\tilde{t}_f\Big] \\ &\quad\coloneqq -\frac{q_J}{p_J}+\frac{1}{p_J}J\Big[\boldsymbol{P}_x\tilde{\boldsymbol{x}}(\cdot)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(\cdot)+\boldsymbol{q}_u,p_t\tilde{t}_0+q_t,p_t\tilde{t}_f+q_t\Big] \\ &\quad= -\frac{q_J}{p_J}+\frac{1}{p_J}E\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}_0+\boldsymbol{q}_x,\boldsymbol{P}_x\tilde{\boldsymbol{x}}_f+\boldsymbol{q}_x,p_t\tilde{t}_0+q_t,p_t\tilde{t}_f+q_t\Big) \\ &\quad+\frac{p_t}{p_J}\int_{\tilde{t}_0}^{\tilde{t}_f}F\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big)\,\mathrm{d}\tilde{t} \end{aligned} \tag{7}$$

where t in the integrand in Eq. (7) is to be understood as $(p_t\tilde{t}+q_t)$.

The transformation of the dynamics is given by

$$\frac{d\boldsymbol{x}}{dt}=\frac{\boldsymbol{P}_x}{p_t}\frac{d\tilde{\boldsymbol{x}}}{d\tilde{t}}=\boldsymbol{f}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big)$$

Hence, the dynamical equations transform according to

$$\begin{aligned} \frac{d\tilde{\boldsymbol{x}}}{d\tilde{t}} &= \tilde{\boldsymbol{f}}\Big(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{u}}(t),\tilde{t}\Big) \\ &\coloneqq p_t\boldsymbol{P}_x^{-1}\boldsymbol{f}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big) \end{aligned} \tag{8}$$

where we have once again used t in Eq. (8) to mean $(p_t\tilde{t}+q_t)$.

By using similar procedures, it follows that the endpoint and path constraint functions transform according to

$$\begin{aligned} \tilde{\boldsymbol{e}}\Big(\tilde{\boldsymbol{x}}_0,\tilde{\boldsymbol{x}}_f,\tilde{t}_0,\tilde{t}_f\Big) &= \boldsymbol{P}_e^{-1}\Big[\boldsymbol{e}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}_0+\boldsymbol{q}_x,\boldsymbol{P}_x\tilde{\boldsymbol{x}}_f+\boldsymbol{q}_x,p_t\tilde{t}_0 \\ &\quad+q_t,p_t\tilde{t}_f+q_t\Big)-\boldsymbol{q}_e\Big] \end{aligned} \tag{9}$$

$$\tilde{\boldsymbol{h}}\Big(\tilde{\boldsymbol{x}}(\tilde{t}),\tilde{\boldsymbol{u}}(\tilde{t}),\tilde{t}\Big)=\boldsymbol{P}_h^{-1}\Big[\boldsymbol{h}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big)-\boldsymbol{q}_h\Big] \tag{10}$$

The corresponding lower and upper bounds are given by

$$\tilde{\boldsymbol{e}}^L=\boldsymbol{P}_e^{-1}\big(\boldsymbol{e}^L-\boldsymbol{q}_e\big),\qquad \tilde{\boldsymbol{e}}^U=\boldsymbol{P}_e^{-1}\big(\boldsymbol{e}^U-\boldsymbol{q}_e\big) \tag{11}$$

$$\tilde{\boldsymbol{h}}^L=\boldsymbol{P}_h^{-1}(\boldsymbol{h}^L-\boldsymbol{q}_h),\qquad \tilde{\boldsymbol{h}}^U=\boldsymbol{P}_h^{-1}(\boldsymbol{h}^U-\boldsymbol{q}_h) \tag{12}$$

Equations (7–12) constitute problem $\tilde{B}$.

IV. Necessary Conditions for the Scaled Problem

Let $\tilde{\boldsymbol{\lambda}}$, $\tilde{\boldsymbol{\mu}}$, and $\tilde{\boldsymbol{\nu}}$ be the adjoint, path, and endpoint covectors, respectively, associated with the necessary conditions for problem $\tilde{B}$. Then, it follows that the Hamiltonian, the Lagrangian of the Hamiltonian, and the endpoint Lagrangian for problem $\tilde{B}$ are given by [4] the following:

1) The Hamiltonian $\tilde{H}$ is

$$\begin{aligned} \tilde{H}\Big(\tilde{\boldsymbol{\lambda}},\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}},\tilde{t}\Big) &\coloneqq \frac{p_t}{p_J}F\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big) \\ &\quad+p_t\tilde{\boldsymbol{\lambda}}^T\boldsymbol{P}_x^{-1}\boldsymbol{f}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big) \end{aligned} \tag{13}$$

2) The Lagrangian of the Hamiltonian $\bar{\tilde{H}}$ is

$$\begin{aligned} \bar{\tilde{H}}\Big(\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\lambda}},\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}},\tilde{t}\Big) &\coloneqq \tilde{H}\Big(\tilde{\boldsymbol{\lambda}},\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}},\tilde{t}\Big) \\ &\quad+\tilde{\boldsymbol{\mu}}^T\boldsymbol{P}_h^{-1}\Big[\boldsymbol{h}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}(t)+\boldsymbol{q}_x,\boldsymbol{P}_u\tilde{\boldsymbol{u}}(t)+\boldsymbol{q}_u,p_t\tilde{t}+q_t\Big)-\boldsymbol{q}_h\Big] \end{aligned} \tag{14}$$

3) The endpoint Lagrangian $\bar{\tilde{E}}$ is

$$\begin{aligned} &\bar{\tilde{E}}\Big(\tilde{\boldsymbol{\nu}},\tilde{\boldsymbol{x}}_0,\tilde{\boldsymbol{x}}_f,\tilde{t}_0,\tilde{t}_f\Big) \\ &\quad\coloneqq \frac{1}{p_J}E\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}_0+\boldsymbol{q}_x,\boldsymbol{P}_x\tilde{\boldsymbol{x}}_f+\boldsymbol{q}_x,p_t\tilde{t}_0+q_t,p_t\tilde{t}_f+q_t\Big) \\ &\quad+\tilde{\boldsymbol{\nu}}^T\boldsymbol{P}_e^{-1}\Big[\boldsymbol{e}\Big(\boldsymbol{P}_x\tilde{\boldsymbol{x}}_0+\boldsymbol{q}_x,\boldsymbol{P}_x\tilde{\boldsymbol{x}}_f+\boldsymbol{q}_x,p_t\tilde{t}_0+q_t,p_t\tilde{t}_f+q_t\Big)-\boldsymbol{q}_e\Big] \end{aligned} \tag{15}$$

The adjoint equation for problem $\tilde{B}$ is given by

$$-\frac{d\tilde{\boldsymbol{\lambda}}}{d\tilde{t}}=\frac{\partial\bar{\tilde{H}}}{\partial\tilde{\boldsymbol{x}}} \tag{16}$$

Evaluating the right-hand side of Eq. (16) using Eq. (14), we get

$$-\frac{d\tilde{\boldsymbol{\lambda}}}{d\tilde{t}}=\frac{p_t}{p_J}\boldsymbol{P}_x^T\left[\frac{\partial F}{\partial\boldsymbol{x}}\right]+p_t\left[\boldsymbol{P}_x^{-1}\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{x}}\boldsymbol{P}_x\right]^T\tilde{\boldsymbol{\lambda}}+\left[\boldsymbol{P}_h^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{x}}\boldsymbol{P}_x\right]^T\tilde{\boldsymbol{\mu}} \tag{17}$$

Following the same process for the stationarity condition associated with the Hamiltonian minimization condition for problem $\tilde{B}$, we get

$$\begin{aligned} \frac{\partial\bar{\tilde{H}}}{\partial\tilde{\boldsymbol{u}}} &= \frac{p_t}{p_J}\boldsymbol{P}_u^T\left(\frac{\partial F}{\partial\boldsymbol{u}}\right)+p_t\boldsymbol{P}_u^T\left(\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{u}}\right)^T(\boldsymbol{P}_x^{-1})^T\tilde{\boldsymbol{\lambda}}+\boldsymbol{P}_u^T\left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{u}}\right)^T(\boldsymbol{P}_h^{-1})^T\tilde{\boldsymbol{\mu}} \\ &= \boldsymbol{0} \end{aligned} \tag{18}$$

Likewise, the initial and final transversality conditions are given by

$$-\tilde{\boldsymbol{\lambda}}(\tilde{t}_0)=\frac{\partial\bar{\tilde{E}}}{\partial\tilde{\boldsymbol{x}}_0}=\frac{\boldsymbol{P}_x^T}{p_J}\frac{\partial E}{\partial\boldsymbol{x}_0}+\left[\boldsymbol{P}_e^{-1}\frac{\partial\boldsymbol{e}}{\partial\boldsymbol{x}_0}\boldsymbol{P}_x\right]^T\tilde{\boldsymbol{\nu}} \tag{19a}$$

$$\tilde{\boldsymbol{\lambda}}(\tilde{t}_f)=\frac{\partial\bar{\tilde{E}}}{\partial\tilde{\boldsymbol{x}}_f}=\frac{\boldsymbol{P}_x^T}{p_J}\frac{\partial E}{\partial\boldsymbol{x}_f}+\left[\boldsymbol{P}_e^{-1}\frac{\partial\boldsymbol{e}}{\partial\boldsymbol{x}_f}\boldsymbol{P}_x\right]^T\tilde{\boldsymbol{\nu}} \tag{19b}$$

Finally, the complementarity conditions are given by $\tilde{\boldsymbol{\mu}}\dagger\tilde{\boldsymbol{h}}$ and $\tilde{\boldsymbol{\nu}}\dagger\tilde{\boldsymbol{e}}$.

V. Balancing Conditions

Proposition A: Let $\tilde{\boldsymbol{x}}^*(\cdot)$, $\tilde{\boldsymbol{u}}^*(\cdot)$, $\tilde{t}_0^*$, and $\tilde{t}_f^*$ be an extremal solution to the scaled problem $\tilde{B}$. Let $\tilde{\boldsymbol{\lambda}}^*(\cdot)$, $\tilde{\boldsymbol{\mu}}^*(\cdot)$, and $\tilde{\boldsymbol{\nu}}^*$ be a multiplier triple associated with this extremal solution. Let $\boldsymbol{P}_\lambda$, $\boldsymbol{P}_\mu$, and $\boldsymbol{P}_\nu$ be invertible matrices defined according to

$$\boldsymbol{P}_\lambda \coloneqq p_J[\boldsymbol{P}_x^{-1}]^T \tag{20a}$$

$$\boldsymbol{P}_\mu \coloneqq \frac{p_J}{p_t}[\boldsymbol{P}_h^{-1}]^T \tag{20b}$$

$$\boldsymbol{P}_\nu \coloneqq p_J[\boldsymbol{P}_e^{-1}]^T \tag{20c}$$

Then, an extremal solution to the unscaled problem B exists and is given by

$$\boldsymbol{x}^*(\cdot) \coloneqq \boldsymbol{P}_x\tilde{\boldsymbol{x}}^*(\cdot) + \boldsymbol{q}_x \tag{21a}$$

$$\boldsymbol{u}^*(\cdot) \coloneqq \boldsymbol{P}_u\tilde{\boldsymbol{u}}^*(\cdot) + \boldsymbol{q}_u \tag{21b}$$

$$t \coloneqq p_t\tilde{t} + q_t \tag{21c}$$

$$\boldsymbol{\lambda}^*(\cdot) \coloneqq \boldsymbol{P}_\lambda\tilde{\boldsymbol{\lambda}}^*(\cdot) \tag{21d}$$

$$\boldsymbol{\mu}^*(\cdot) \coloneqq \boldsymbol{P}_\mu\tilde{\boldsymbol{\mu}}^*(\cdot) \tag{21e}$$

$$\boldsymbol{\nu}^* \coloneqq \boldsymbol{P}_\nu\tilde{\boldsymbol{\nu}}^* \tag{21f}$$

Proof: The proof of Eqs. (21a–21c) follows quite simply by construction. The proof (21d–21f) follows by substituting Eq. (21) in problem B^λ and using Eq. (20) to show that the resulting equations are the same as the necessary conditions for the scaled problem derived in the previous section.

A. Remark 1

Once $\boldsymbol{P}_x$, $\boldsymbol{P}_h$, and $\boldsymbol{P}_e$ are chosen to scale the primal variables and constraints, then, according to Proposition *A*, there exist dual variables $\boldsymbol{\lambda}^*(\cdot)$, $\boldsymbol{\mu}^*(\cdot)$, and $\boldsymbol{\nu}^*$ that get scaled automatically in compliance with Eq. (21). Consequently, balancing can now be defined more precisely as choosing $\boldsymbol{P}_x$, $\boldsymbol{P}_h$ and $\boldsymbol{P}_e$ along with p_J and p_t such that the values of the covectors $\boldsymbol{\lambda}^*(\cdot)$, $\boldsymbol{\mu}^*(\cdot)$, and $\boldsymbol{\nu}^*$ are of similar orders of magnitude as their corresponding vectors. In contrast, scaling is choosing $\boldsymbol{P}_x$, $\boldsymbol{P}_h$ and $\boldsymbol{P}_e$ such that the values of the corresponding vectors (as well as their components) are of similar magnitude relative to each other. When both requirements are met, the problem is said to be scaled and balanced.

B. Remark 2

Substituting Eq. (20) in Eq. (13), it is clear that the value of the Hamiltonian transforms according to

$$H(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}, t) = \left(\frac{p_J}{p_t}\right)\tilde{H}\left(\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}, \tilde{t}\right) \tag{22}$$

C. Remark 3

A natural choice for $\boldsymbol{P}_x$, $\boldsymbol{P}_h$, and $\boldsymbol{P}_e$ is diagonal matrices. For these choices, $\boldsymbol{P}_\lambda$, $\boldsymbol{P}_\mu$, and $\boldsymbol{P}_\nu$ are also diagonal matrices. In this situation, each component of a vector is independently related to each component of its corresponding covector. This fact can be used in a numerical setting for a simple algorithmic technique for scaling and balancing.

D. Remark 4

Let the units of $\boldsymbol{x}$, $\boldsymbol{h}$, and $\boldsymbol{e}$ be given according to

$$\boldsymbol{x} \coloneqq \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N_x} \end{bmatrix} \begin{matrix} x_1 \text{ units} \\ x_2 \text{ units} \\ \vdots \\ x_{N_x} \text{ units} \end{matrix} \tag{23a}$$

$$\boldsymbol{h} \coloneqq \begin{bmatrix} h_1(\boldsymbol{x}, \boldsymbol{u}, t) \\ h_2(\boldsymbol{x}, \boldsymbol{u}, t) \\ \vdots \\ h_{N_h}(\boldsymbol{x}, \boldsymbol{u}, t) \end{bmatrix} \begin{matrix} h_1 \text{ units} \\ h_2 \text{ units} \\ \vdots \\ h_{N_h} \text{ units} \end{matrix} \tag{23b}$$

$$\boldsymbol{e}(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) \coloneqq \begin{bmatrix} e_1(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) \\ e_2(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) \\ \vdots \\ e_{N_e}(\boldsymbol{x}_0, \boldsymbol{x}_f, t_0, t_f) \end{bmatrix} \begin{matrix} e_1 \text{ units} \\ e_2 \text{ units} \\ \vdots \\ e_{N_e} \text{ units} \end{matrix} \tag{23c}$$

Note that no assumption is made on any of the units in Eq. (23). Thus, for example, x_1 units may be meters, and x_2 units may be feet, even if the pair (x_1, x_2) is the position coordinate of the same point mass. Obviously, the Euclidian norm of the numbers given by x_1 and x_2 has no physical meaning. In the same spirit, x_3 units may be yards per day, even if the variable x_3 is the time-rate change of x_1. Despite such arbitrary choices, it is still possible to measure a length of a vector through the use of a covector. To this end, we let $\boldsymbol{P}_x$, $\boldsymbol{P}_h$, and $\boldsymbol{P}_e$ be diagonal matrices. Then, it follows from Eqs. (20) and (21) that the covectors have units given by

$$\boldsymbol{\lambda} \coloneqq \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{N_x} \end{bmatrix} \begin{matrix} \text{CU}/x_1 \text{ units} \\ \text{CU}/x_2 \text{ units} \\ \vdots \\ \text{CU}/x_{N_x} \text{ units} \end{matrix} \tag{24a}$$

$$\boldsymbol{\mu} \coloneqq \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_{N_h} \end{bmatrix} \begin{matrix} \frac{\text{CU/TU}}{h_1 \text{ units}} \\ \frac{\text{CU/TU}}{h_2 \text{ units}} \\ \vdots \\ \frac{\text{CU/TU}}{h_{N_h} \text{ units}} \end{matrix} \tag{24b}$$

$$\boldsymbol{\nu} \coloneqq \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_{N_e} \end{bmatrix} \begin{matrix} \text{CU}/e_1 \text{ units} \\ \text{CU}/e_2 \text{ units} \\ \vdots \\ \text{CU}/e_{N_e} \text{ units} \end{matrix} \tag{24c}$$

where CU is the cost unit and TU is the time unit. Equation (24) was first introduced in [4] as part of the definition of a covector associated with the relevant vector. Note also from Eq. (24) that a covector always has some unit of frequency; i.e., it is always given in terms of some common unit per some unit. The common unit in Eqs. (24a) and (24c) is the cost unit, but the common unit in Eq. (24b) is the cost unit per time unit. Consequently, we can now measure the length of a vector by an appropriate covector. For instance, $\boldsymbol{x}^T\boldsymbol{\lambda}$ generates a scalar in terms of CUs despite that its Euclidean norm might not be computable due to the disparity in units of the constituents of $\boldsymbol{x}$. See [4], Sec. 2.2, for further details.

E. Remark 5

As a consequence of Eqs. (23) and (24), scaling may be conceived as simply changing units. Consequently, nondimensionalization is also changing units and should not be construed as eliminating units.

F. Remark 6

The Hamiltonian is not dimensionless. The unit of the Hamiltonian is the cost unit per time unit. This follows from Eq. (24). When the cost unit is the same as the time unit (e.g., time optimality), then, and only then, is the Hamiltonian truly dimensionless.

G. Remark 7

Using Eq. (8), we can write

$$\frac{\partial \tilde{\boldsymbol{f}}}{\partial \tilde{\boldsymbol{x}}} = p_t \boldsymbol{P}_x^{-1} \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right) \boldsymbol{P}_x \tag{25}$$

Because of the similarity transformation on the right-hand side of Eq. (25), the spectral radii of the Jacobians are related by

$$\rho(\partial_{\tilde{x}} \tilde{\boldsymbol{f}}) = p_t \rho(\partial_x \boldsymbol{f}) \tag{26}$$

where $\rho(\cdot)$ is the spectral radius of $(\cdot)$. Rewriting p_t as $(t_f - t_0)/(\tilde{t}_f - \tilde{t}_0)$ and substituting in Eq. (26), we get an invariance equation,

$$(\tilde{t}_f - \tilde{t}_0)\rho(\partial_{\tilde{x}} \tilde{\boldsymbol{f}}) = (t_f - t_0)\rho(\partial_x \boldsymbol{f}) \tag{27}$$

Because the product of the spectral radius and the time horizon is a key sensitivity factor (see [4], Sec. 2.9), it follows from Eq. (27) that the curse of sensitivity cannot be mitigated by scaling.

VI. Example Illustrating Designer Units, Scaling, and Balancing

Although the concepts of designer units, scaling, and balancing have been used for more than a decade to generate successful flight and field operations [9–12,14,17,20], the process has largely been ad hoc until recently. Because the flight problems are too involved for illustrating the procedures, we design a far simpler problem that typifies the systematic process of scaling and balancing. To this end, consider the well-known Brachistochrone problem. Instead of using well-scaled numbers as is customarily discussed in many textbooks, we purposefully choose the final-time condition on the position coordinates (x, y) to be widely disparate and given by (1000,1) m; see Fig. 1. This results in a badly scaled or bad Brachistochrone problem that can be formulated as [4]

$$(B_R)\begin{cases} \left.\begin{aligned} \mathbb{X} = \mathbb{R}^3 \qquad & \mathbb{U} = \mathbb{R} \\ \boldsymbol{x} = (x, y, v) \quad & \boldsymbol{u} = \theta \end{aligned}\right\} \text{(preamble)} \\ \text{Minimize} \qquad J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), t_f] = t_f \quad \} \text{(cost)} \\ \text{Subject to} \\ \left.\begin{aligned} \dot{x} &= v \sin\theta \\ \dot{y} &= v \cos\theta \\ \dot{v} &= g \cos\theta \end{aligned}\right\} \text{(dynamics)} \\ \left.\begin{aligned} (t_0, x_0, y_0, v_0) &= (0, 0, 0, 0) \\ (x_f, y_f) &= (1000, 1) \end{aligned}\right\} \text{(endpoints)} \end{cases}$$

where $g = 9.8$ m/s^2. See Fig. 1 for a physical definition of the variables.

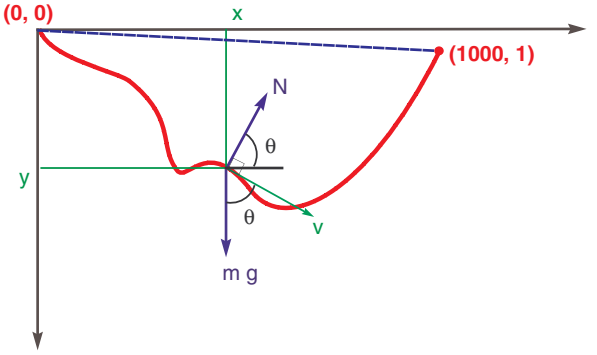

Fig. 1 Schematic for the bad Brachistochrone problem. Figure is not to scale.

A. Illustrating the Process for Choosing Designer Units

As a consequence of Eq. (23a), we can write

$$\boldsymbol{x} := \begin{bmatrix} x \\ y \\ v \end{bmatrix} \begin{matrix} x \text{ units} \\ y \text{ units} \\ v \text{ units} \end{matrix} \tag{28}$$

Because the cost unit is the same as the time unit, the adjoint covector is defined by

$$\boldsymbol{\lambda} := \begin{bmatrix} \lambda_x \\ \lambda_y \\ \lambda_v \end{bmatrix} \begin{matrix} t \text{ units}/x \text{ units} \\ t \text{ units}/y \text{ units} \\ t \text{ units}/v \text{ units} \end{matrix} \tag{29}$$

As a result, the Hamiltonian

$$H(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}) = \lambda_x v \sin\theta + \lambda_y v \cos\theta + \lambda_v g \cos\theta \tag{30}$$

is dimensionless.

Applying the Hamiltonian minimization condition, we get

$$\lambda_x(t)v(t)\cos\theta(t) - \lambda_y(t)v(t)\sin\theta(t) - \lambda_v(t)g\sin\theta(t) = 0 \quad (\forall\ t \in [t_0, t_f]) \tag{31}$$

The adjoint equations,

$$\begin{aligned} \dot{\lambda}_x &= 0 \\ \dot{\lambda}_y &= 0 \\ \dot{\lambda}_v &= -\lambda_x \sin\theta - \lambda_y \cos\theta \end{aligned} \tag{32}$$

indicate that λ_x and λ_y are constants. In addition, the transversality condition,

$$\lambda_v(t_f) = 0 \tag{33}$$

and the Hamiltonian value condition,

$$H[@t_f] := \lambda_x(t_f)v(t_f)\sin\theta(t_f) + \lambda_y(t_f)v(t_f)\cos\theta(t_f) + \lambda_v(t_f)g\cos\theta(t_f) = -1 \tag{34}$$

complete the computational set of conditions that define the boundary value problem. These equations can also be used as part of the totality of a verification and validation of a candidate optimal solution obtained by any computational method.

To perform initial scaling, we simply take the given numerical data as a starting point. For the numerics given in problem B_R, we expect x to satisfy

$$0 \le x \le 1000 \tag{35}$$

In the absence of further analysis, we can assume y to take on a similar range of values. Furthermore, because the x distance is relatively large, we expect the time of travel to be relatively large (in terms of seconds). Based on these heuristics, we choose an initial set of scaling factors according to

$$\begin{aligned} x &= P_x\tilde{x} = 100\tilde{x} \\ y &= P_y\tilde{y} = 20\tilde{y} \\ v &= P_v\tilde{v} = 10\tilde{v} \\ \theta &= P_\theta\tilde{\theta} = \tilde{\theta} \\ t &= p_t\tilde{t} = 10\tilde{t} \end{aligned} \tag{36}$$

As a result of Eq. (36), the large variation in x indicated by Eq. (35) is tempered by $\tilde{x}$ according to

$$0 \le \tilde{x} \le 10$$

Note that the numbers given in Eq. (36) imply a unit of distance along the x axis that is completely different from the unit of distance along the y axis. In fact, these numbers constitute a specific system of units that do not conform with the metric system or any other set of the standard units; hence, these are designer units. The conversion between the designer units of Eq. (36) and the metric units is given by

$$\begin{aligned} 1\tilde{x}\,\text{unit} &= 1\,\text{unit of distance along } x\,\text{axis} = 100\ \text{m} \\ 1\tilde{y}\,\text{unit} &= 1\,\text{unit of distance along } y\,\text{axis} = 20\ \text{m} \\ 1\tilde{v}\,\text{unit} &= 1\,\text{unit of speed} = 10\ \text{m/s} \\ 1\tilde{t}\,\text{unit} &= 1\,\text{unit of time} = 10\ \text{s} \end{aligned} \tag{37}$$

It is important to note that the velocity unit is completely independent of any of the x, y, or t units. Consequently, these designer units are not consistent in the sense that

$$\frac{\mathrm{d}\tilde{y}}{\mathrm{d}\tilde{t}} \neq \tilde{v}\cos\tilde{\theta}$$

To drive home this point, we note that we can no longer express g in terms of distance units per the square of time units. For instance, in the metric system, the unit of g is given by meters per second squared. Because we chose distance units along the x and y directions to be independent of each other, it is clear that we cannot regard g in terms of distance units per the square of time units. The proper unit for g is obtained by considering $\mathrm{d}\tilde{v}/\mathrm{d}\tilde{t}$. This implies that we may regard the gravitational acceleration as being transformed according to

$$\tilde{g} = \left(\frac{p_t}{P_v}\right)g = 9.8\tilde{v}\,\text{unit}/\tilde{t}\,\text{unit} \tag{38}$$

That is, $\tilde{g} \neq 1$ numerically, which is a typical number for canonical units [3]. That $\tilde{g} = g$ numerically in Eq. (38) is simply coincidental and is a result of choosing $p_t = P_v$ in Eq. (36).

Using the scaling units of Eq. (36), the endpoint conditions can be written as

$$\begin{aligned} \tilde{t}_0 &= 0 && && (\text{in } \tilde{t}\ \text{units}) \\ \tilde{x}(\tilde{t}_0) &= 0 && \tilde{x}(\tilde{t}_f) = 10 && (\text{in } \tilde{x}\ \text{units}) \\ \tilde{y}(\tilde{t}_0) &= 0 && \tilde{y}(\tilde{t}_f) = 0.05 && (\text{in } \tilde{y}\ \text{units}) \\ \tilde{v}_0 &= 0 && && (\text{in } \tilde{v}\ \text{units}) \end{aligned} \tag{39}$$

Imposing the endpoint conditions according to Eq. (39) is tantamount to choosing $\boldsymbol{P}_e$ according to

$$\boldsymbol{P}_e = \text{diag}(10, 100, 100, 20, 20, 10) \tag{40}$$

This follows as a direct consequence of Eq. (36). Finally, we scale the cost functional using $p_J = 10$ so that we can write

$$\tilde{J}\left[\tilde{\boldsymbol{x}}(\cdot), \tilde{\boldsymbol{u}}(\cdot), \tilde{t}_0, \tilde{t}_f\right] = \tilde{t}_f$$

At this juncture, we wish to emphasize that the purpose of initial scaling is not necessarily to generate the best set of designer units; rather, it is largely directed at producing a work flow for balancing. As will be apparent shortly, once an initial numerical result is obtained, balancing can usually be performed in just about two iterations. One situation in which initial scaling becomes critical to the work flow is when no numerical result is achieved simply because of poor scaling. The detection and mitigation of this problem are open areas of research.

B. Illustrating the Process of Descaling Covectors

For the purposes of clarity of the discussions to follow, we use the following terminology:

1) *Unscaled* refers to all numbers and variables associated with the original (or unscaled) problem.

2) *Scaled* refers to all numbers and variables associated with the affinely transformed (or scaled) problem.

3) *Descaled* refers to all numbers and variables that are purported solutions to the unscaled problem obtained via Eq. (21) and a solution to the scaled problem.

Applying Eq. (21) to descale the adjoint covector, we get

$$\begin{aligned} \lambda_x &= \tilde{\lambda}_x\left(\frac{p_J}{P_x}\right) = \frac{\tilde{\lambda}_x}{10}\ (\text{s/m}) \\ \lambda_y &= \tilde{\lambda}_y\left(\frac{p_J}{P_y}\right) = \frac{\tilde{\lambda}_y}{2}\ (\text{s/m}) \\ \lambda_v &= \tilde{\lambda}_v\left(\frac{p_J}{P_v}\right) = \tilde{\lambda}_v\ (\text{s}^2/\text{m}) \end{aligned} \tag{41}$$

Similarly, from Eqs. (40) and (20), we have

$$\begin{aligned} \boldsymbol{P}_\nu &= 10\left[\boldsymbol{P}_e^{-1}\right]^T \Rightarrow \left(\nu_{t_0}, \nu_{x_0}, \nu_{x_f}, \nu_{y_0}, \nu_{y_f}, \nu_{v_0}\right) \\ &= \left(\tilde{\nu}_{t_0}, \frac{\tilde{\nu}_{x_0}}{10}, \frac{\tilde{\nu}_{x_f}}{10}, \frac{\tilde{\nu}_{y_0}}{2}, \frac{\tilde{\nu}_{y_f}}{2}, \tilde{\nu}_{v_0}\right) \end{aligned} \tag{42}$$

where ν and $\tilde{\nu}$ with the appropriate subscripts are the endpoint multipliers associated with the initial- and final-time conditions. Note that these endpoint multipliers also have units similar to those identified in Eq. (41).

C. Illustrating the Numerical Process of Scaling and Balancing

All of the analysis so far has been agnostic to the specific choice of a numerical method or software. To demonstrate the numerical process, any appropriate mathematical software may be used. We begin by choosing DIDO©, a MATLAB® toolbox for solving optimal-control problems [4]. DIDO is the same tool that was used in all of the flight applications noted earlier. It is based on the spectral algorithm [25–30] for pseudospectral optimal control and does not require any guess of the solution to solve the problem. Furthermore, DIDO automatically generates all of the covectors associated with a generic optimal-control problem (see problem B^λ presented in Sec. II of this paper) through an implementation of the covector mapping principle [4,20,30]. That is, DIDO generates a guess-free candidate solution to the BVP while only requiring the data functions for problem B. This is why the spectral algorithm and its implementation in DIDO do not belong to the class of direct or indirect methods. In fact, these ideas effectively obviate the need for such a classification; see Sec. 2.9.2 of [4].

1. Initial Scaling and Descaling

A candidate primal-dual solution (generated by DIDO) is shown in Fig. 2. By definition, these are simply candidate solutions to the scaled problem. They have not yet been validated. Because it is frequently more meaningful to validate results in physical units, we first descale the primal variables using Eq. (36). The descaled candidate control trajectory is then used to propagate the initial conditions,

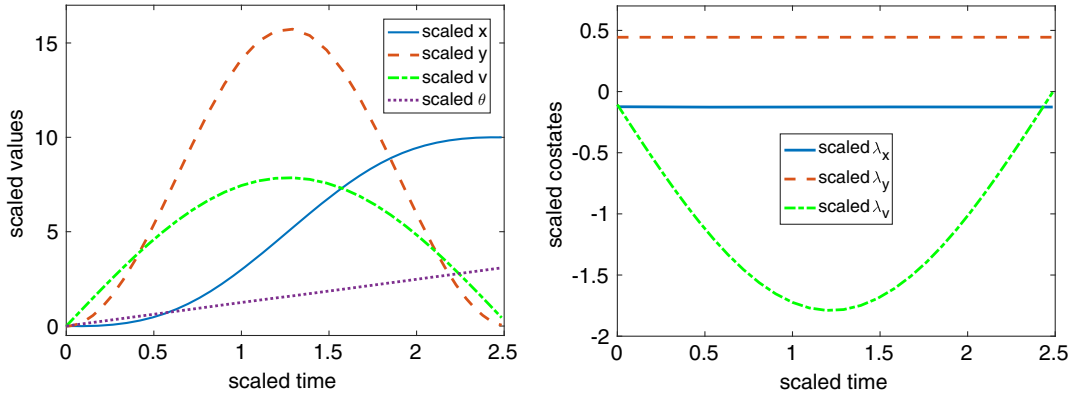

Fig. 2 Guess-free primal (left) and dual (right) solutions to the badly scaled Brachistochrone problem.

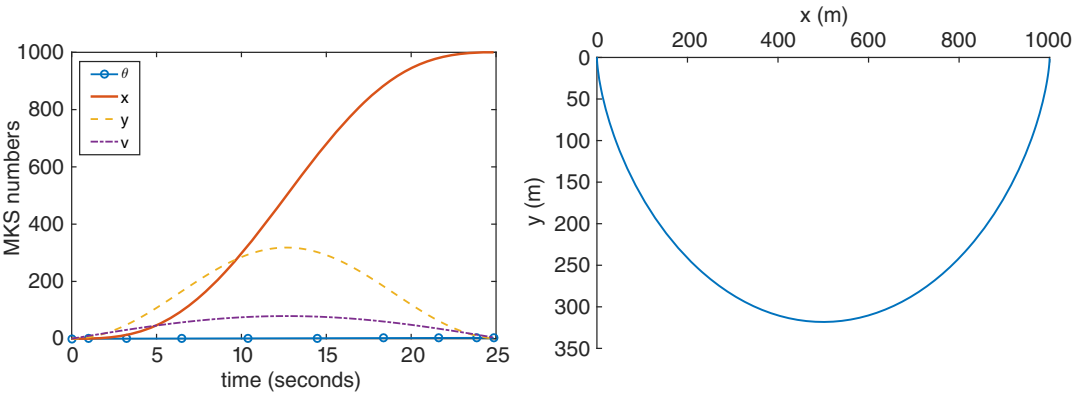

Fig. 3 Primal feasible solution to the badly scaled Brachistochrone problem indicating variations in three orders of magnitude (MKS, metre, kilogram, and/or second).

$$x(0) = 0, \quad y(0) = 0, \quad v(0) = 0$$

using linear interpolation for the controls and ode45 in MATLAB®. The propagated state trajectory is shown in Fig. 3. The propagated values of $x(t_f)$ and $y(t_f)$ satisfy the final-time conditions to the following precision:

$$|x(t_f) - 1000| = 3.6 \times 10^{-3} \text{ m}, \quad |y(t_f) - 1| = 1.9 \times 10^{-4} \text{ m}$$

Thus, the descaled solution is verifiably feasible. In flight applications, such an independent verification of feasibility is critical to a successful preflight checkout [12,14,20].

To validate the extremality of the feasible solution, we use Eqs. (41) and (22) to descale the adjoint covectors and the evolution of the Hamiltonian, respectively. The results are shown in Fig. 4. It is apparent that $\lambda_x(t)$ and $\lambda_y(t)$ are constants as required by Eq. (32). It is also apparent that the Hamiltonian is nearly a constant and equal to -1 as required by Eq. (34) and the first integral. Thus, the theoretical necessary conditions are satisfied up to the indicated approximations.¶ These indicators of optimality validate that the solution presented in Fig. 2 is at least an extremal.

¶Although the spectral algorithm can theoretically generate very accurate solutions [4,20,28,30], the Hamiltonian evolution equation is satisfied only weakly [36]; hence, the Hamiltonian is not expected to be equal to -1 in the strong L_∞ norm in Fig. 4. In addition, because no Jacobian information was provided in the generation of Fig. 4, the accuracy in the computation of the dual variables is expected to be lower than that of the primal solution. Note also that dual (or primal) tolerances cannot be set to arbitrarily small numbers (e.g., 10^{-6} or 10^{-8}) to attain higher accuracy because they may violate consistency conditions [31]. See [32] for details and [33] for a unified framework.

2. *Illustrating Universality of Proposition A*

To illustrate that Proposition A is indeed universal (and not merely specific to DIDO), we construct a shooting algorithm using ode45 and fsolve from the MATLAB® optimization toolbox. The objective of ode45 is to generate the vector function,

$$\boldsymbol{S}\colon \left(\lambda_{x_0}, \lambda_{y_0}, \lambda_{v_0}, t_f\right) \mapsto \left(x_f, y_f, v_f, H_f\right) \tag{43}$$

by integrating the six state-costate equations using the initial conditions (at $t_0 = 0$),

$$x(t_0) = 0, \quad y(t_0) = 0, \quad v(t_0) = 0,$$
$$\lambda_x(t_0) = \lambda_{x_0}, \quad \lambda_y(t_0) = \lambda_{y_0}, \quad \lambda_v(t_0) = \lambda_{v_0}$$

The quantity H_f in Eq. (43) is the final value of the Hamiltonian evaluated using the results of the integration and Eq. (30). The objective of fsolve is to solve for the zeros of the residual vector function $\boldsymbol{r}\colon \mathbb{R}^4 \to \mathbb{R}^4$ defined by

$$\boldsymbol{r}\left(\lambda_{x_0}, \lambda_{y_0}, \lambda_{v_0}, t_f\right) := \boldsymbol{S}(\lambda_{x_0}, \lambda_{y_0}, \lambda_{v_0}, t_f) - \begin{pmatrix} 1000 \\ 1 \\ 0 \\ -1 \end{pmatrix} \tag{44}$$

Because a shooting algorithm is fundamentally doomed by the curse of sensitivity [4], we choose the values of the guess to be almost exactly equal to the expected solution,

$$\lambda_{x_0} = -0.013, \quad \lambda_{y_0} = 0.225, \quad \lambda_{v_0} = -0.113, \quad t_f = 24.0 \tag{45}$$

For the purposes of brevity, we limit our discussions to only the costate trajectories. Shown in Fig. 5 are both the unscaled and scaled costates trajectories obtained by the shooting algorithm. The scaled costates were obtained by using the scaled equations and replacing the numerical value of the 4-vector in Eq. (44) by its scaled

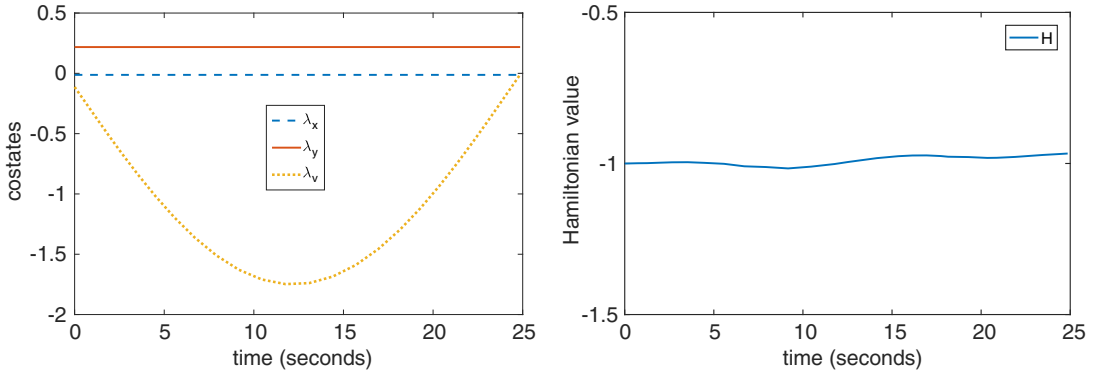

Fig. 4 Descaled costates and the evolution of the Hamiltonian for the badly scaled Brachistochrone problem.

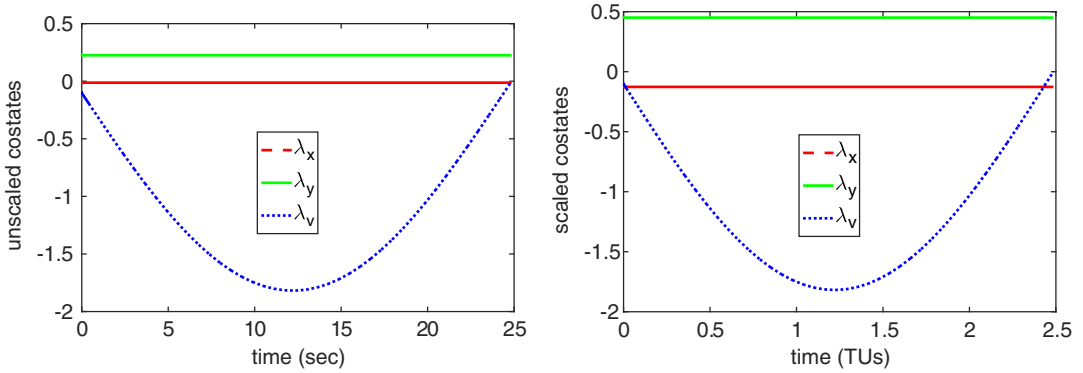

Fig. 5 Unscaled and scaled costates obtained by a shooting method for the badly scaled Brachistochrone problem (TUs, time unit).

counterpart [see Eq. (39)]. It is apparent that the unscaled costates match the descaled costates (see Fig. 4) and the scaled costates match the DIDO result shown in Fig. 2 to numerical precision. In other words, we have demonstrated that Proposition A is independent of the numerical algorithm or software.

3. *Illustrating a Process for Better Balancing*

As a final point of illustration, we now demonstrate that it is possible to achieve a more balanced computational optimal control problem. First, note from the range of values of the ordinates in Fig. 2 that the computational problem is not perfectly balanced. This is precisely what happens in solving many flight application problems; that is, it is frequently not possible to choose designer units that achieve well-balanced equations at the first attempt. Nonetheless, after further analysis of the type illustrated in the preceding paragraphs, it is possible to achieve a better balanced computational problem by merely inspecting the results. Based on the range of values indicated in Fig. 2, we now rescale the primal problem using the following units:

$$
\begin{aligned}
x &= P_x\tilde{x} = 1000\tilde{x} \Rightarrow 1 \text{ distance unit along } x \text{ axis} = 1000 \text{ m}\\
y &= P_y\tilde{y} = 160\tilde{y} \Rightarrow 1 \text{ distance unit along } y \text{ axis} = 160 \text{ m}\\
v &= P_v\tilde{v} = 20\tilde{v} \Rightarrow 1 \text{ speed unit} = 20 \text{ m/s}
\end{aligned} \tag{46}
$$

All other choices of units are the same as before; see Eq. (36). Clearly, $\sqrt{\tilde{x}^2 + \tilde{y}^2}$ is meaningless. Note also that the numerical choice of these scaling factors furthers the disparity between the new set of designer units and the original physical units. For instance, the gravitational acceleration transforms according to

$$
\tilde{g} = \left(\frac{p_t}{P_v}\right) g = 4.9\,\tilde{v}\,\text{unit}/\tilde{t}\,\text{unit} \tag{47}
$$

The primal and dual trajectories for the rescaled problem are shown in Fig. 6. By inspection, it is clear that the problem is reasonably well balanced with all variables contained in the range $[-4, 4]$.

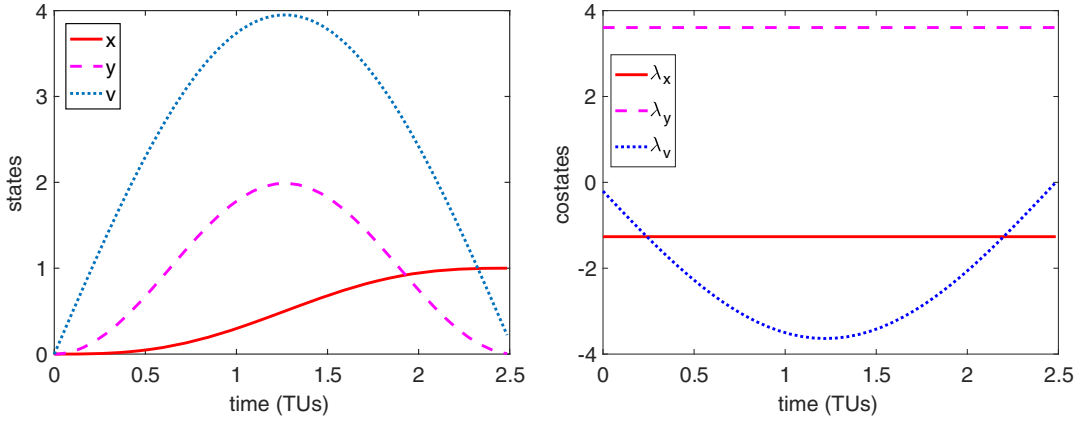

Fig. 6 Rescaled solution to the badly scaled Brachistochrone problem obtained by using the better-balanced set of designer units given by Eq. (46).

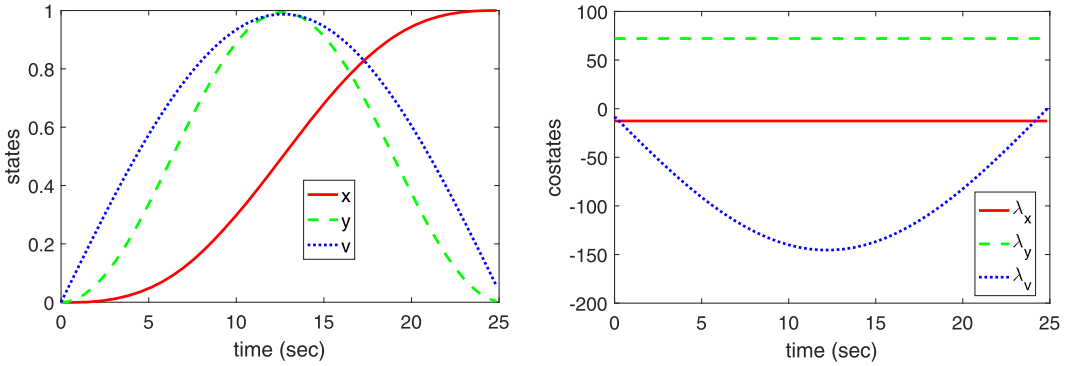

Fig. 7 Primal-normalized solution to the badly scaled Brachistochrone problem obtained by using the set of designer units given by Eq. (48).

4. Illustrating the Fallacy of Balancing on a Unit Interval

Suppose we scale the primal problem using the following designer units:

$$\begin{aligned} x &= P_x\tilde{x} = 1000\tilde{x} \\ y &= P_y\tilde{y} = 320\tilde{y} \\ v &= P_v\tilde{v} = 80\tilde{v} \end{aligned} \tag{48}$$

A quick examination of Fig. 3 shows that Eq. 48 will force all the state variables to lie on the unit interval [0, 1]. That this is indeed the case is shown in Fig. 7. Also shown in Fig. 7 are the corresponding costates. As expected (by Proposition *A* and by inspection of Fig. 4) the dual variables now lie in the range [−200, 100]. In this particular situation, the range [−200, 100] is only about two orders of magnitude away from the desirable interval of [−1, 1]; hence, it is no cause for serious alarm. Nonetheless, it is apparent that the imbalance may be greater in other applications.

In certain applications, the state variables may be naturally constrained to lie on the unit interval (e.g., quaternion parameterization). If the costates (e.g., coquaternions) are imbalanced, it is still possible to balance the state-costate pair by scaling the states to lie on a nonunit interval through a proper selection of $\boldsymbol{P}_x$ in Proposition A. In other words, there is no real reason to be constrained on a unit interval. In stronger terms, the conventional wisdom of scaling on a unit interval is not necessarily the best approach to balancing optimal-control problems.

VII. Adjunct Consequences of Scaling and Balancing

The consequences of scaling and balancing go far beyond faster computation of optimal trajectories. When the pseudospectral optimal-control method of the early days [34–40] (circa 1995–2005) is applied to solve the bad Brachistochrone problem; i.e., without using proper scaling or spectral techniques [25,28,30], the approach fails to produce the correct solution. In general, the same is true of Runge–Kutta collocation methods; see [31,41]. If an algorithm does not generate a feasible solution to a problem with a known solution, such as the Brachistochrone problem, then it is a clear failure of the algorithm. Frequently, we use optimal-control techniques to solve hard problems in which we do not know in advance if a solution exists. In such practical situations, it is important to know if the lack of a feasible solution is due to a failure of the algorithm or a genuine nonexistence of a solution. Consequently, over the last decade, proper scaling and balancing have been fundamentally intertwined with the theory and practice of optimal control [1,4,9,11,13,14].

A. Feasibility via Optimization

In many practical applications, there is a need to simply generate feasible solutions. Consequently, scaling and balancing techniques are critical not only for faster optimization but also to answer the more fundamental and difficult theoretical question [23] of the existence of a solution.

As a means to illustrate this aspect of the intertwining between theory and practice, consider the optimal propellant maneuver (OPM) that is actively used [9] in current flight operations of the International Space Station (ISS). The OPM saves NASA 90% propellant in momentum management at a savings of approximately $1 million per maneuver [9,10,12]. This maneuver was designed by Bedrossian after his discovery of the zero propellant maneuver (ZPM) that saves all 100% of propellant [11]. Before Bedrossian's discovery, it was generally thought that it was impossible to dump all of the accumulated momentum without any propellant consumption [42]. In other words, a feasible solution (for 100% propellant savings) was believed to be nonexistent. When the unscaled values of the ISS angular momentum h, the control torque u, and angular velocity ω are used for dynamic optimization, all prior methods investigated by Bedrossian et al. consistently failed to generate a feasible solution. These variables vary as [11]

$$\begin{aligned} -10^4 &\le h \le 10^4 \text{ lb} \cdot \text{ft} \cdot \text{s} \\ -10^2 &\le u \le 10^2 \text{ lb} \cdot \text{ft} \\ -10^{-4} &\le \omega \le 10^{-4} \text{ rad} \cdot \text{s}^{-1} \end{aligned}$$

Consequently, a failure to find a feasible zero-propellant solution was consistent with the pre-ZPM belief of physics. Were it not for proper scaling and balancing, the ZPM might have gone undiscovered. More specifically, when the variables were transformed according to

$$\begin{aligned} h &= 1000\tilde{h} \text{ lb} \cdot \text{ft} \cdot \text{s} \\ u &= 10\tilde{u} \text{ lb} \cdot \text{ft} \\ t &= 1000\tilde{t} \text{ s} \end{aligned} \tag{49}$$

not only were Bedrossian et al. [11] able to find a feasible solution but also several different solutions. In other words, a special choice of designer units converted a hard problem to an easy one. The rest is history [10].

The lessons learned from such successes and similar ones that followed [9,14–17] are codified in the scaling and balancing techniques presented in the preceding sections. More specifically, the last decade has witnessed the use of theoretical optimization principles to determine the practical feasibility of innovative concepts as opposed to the more conventional use of algorithms to optimize a feasible design. In other words, optimal-control theory has been used as a tool to innovate and not merely to optimize.

B. Nonuniqueness of the Costates

Note that Proposition A only asserts the existence of linearly descaled covectors; it does not imply uniqueness. Using the same arguments as that of Proposition A, it is relatively straightforward to show that the costates in an optimal-control problem are not necessarily unique. This is achieved by replacing the linear equations (21d–21f) with their affine counterparts:

$$\boldsymbol{\lambda}^*(\cdot) \coloneqq \boldsymbol{P}_\lambda\tilde{\boldsymbol{\lambda}}^*(\cdot) + \boldsymbol{q}_\lambda(\cdot) \tag{50a}$$

$$\boldsymbol{\mu}^*(\cdot) \coloneqq \boldsymbol{P}_\mu\tilde{\boldsymbol{\mu}}^*(\cdot) + \boldsymbol{q}_\mu(\cdot) \tag{50b}$$

$$\boldsymbol{\nu}^* \coloneqq \boldsymbol{P}_\nu \tilde{\boldsymbol{\nu}}^* + \boldsymbol{q}_\nu \tag{50c}$$

where $\boldsymbol{q}_\lambda(\cdot)$: $t \mapsto \mathbb{R}^{N_x}$, $\boldsymbol{q}_\mu(\cdot)$: $t \mapsto \mathbb{R}^{N_h}$, and $\boldsymbol{q}_\nu \in \mathbb{R}^{N_e}$. Obviously, $\boldsymbol{q}_\lambda(\cdot) \equiv \boldsymbol{0}$, $\boldsymbol{q}_\mu(\cdot) \equiv \boldsymbol{0}$, and $\boldsymbol{q}_\nu = \boldsymbol{0}$ recover Proposition A; however, by substituting Eq. (50) in problem B^λ, it is straightforward to show that it is not necessary for the $\boldsymbol{q}$ multipliers to be trivial. That Lagrange multipliers are not unique is well known in nonlinear programming [43,44]; hence, it seems apparent that this must also be true in optimal-control programming. However, unlike the static case, the possibility of nonunique costates in optimal control is limited by the Lipschitz continuity of $\partial_x \boldsymbol{f}$. Despite this limitation, the conditions for nonuniqueness are relatively mild; see [4], Sec. 4.9, for a complete worked-out example pertaining to the optimal steering of a rigid body.

C. Fallacy of Discrete Scaling

In Eqs. (4) and (5), we deliberately used affine scaling with constant coefficients. Suppose we choose time-varying scaling coefficients; then, the state variable transformation can be written as

$$\boldsymbol{x}(t) = \boldsymbol{P}_x(t)\tilde{\boldsymbol{x}}(t) + \boldsymbol{q}_x(t) \tag{51}$$

Differentiating Eq. (51) and substituting the unscaled dynamics in the resulting equation generates

$$\begin{aligned}\frac{d\tilde{\boldsymbol{x}}}{d\tilde{t}} &= p_t \boldsymbol{P}_x^{-1}(t)\Big(\dot{\boldsymbol{x}}(t) - \dot{\boldsymbol{P}}_x(t)\tilde{\boldsymbol{x}}(t) - \dot{\boldsymbol{q}}_x(t)\Big) \\ &= p_t \boldsymbol{P}_x^{-1}(t)\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) - \underbrace{p_t \boldsymbol{P}_x^{-1}(t)\Big(\dot{\boldsymbol{P}}_x(t)\tilde{\boldsymbol{x}}(t) + \dot{\boldsymbol{q}}_x(t)\Big)}_{\text{additional dynamics}}\end{aligned} \tag{52}$$

That is, the transformed state dynamics contain additional dynamics. Consequently, it generates the following questions:

1) How do we choose $\dot{\boldsymbol{P}}_x(t)$ and $\dot{\boldsymbol{q}}_x(t)$? That is, what is the rationale for choosing these functions?

2) Because our objective is to generate a theory for generic problems (e.g., generic dynamics), how can we choose universal functions $\dot{\boldsymbol{P}}_x(t)$ and $\dot{\boldsymbol{q}}_x(t)$?

3) Even if we were to severely limit time-varying scaling to a specific dynamical system, how do we choose $\dot{\boldsymbol{P}}_x(t)$ and $\dot{\boldsymbol{q}}_x(t)$ whose properties remain valid for all feasible control functions $\boldsymbol{u}(\cdot)$?

4) How do we ensure that the additional dynamics indicated in Eq. (52) does not create new numerical problems over the space of all differentiable functions $\boldsymbol{f}$?

From these basic considerations, it is clear that time-varying affine scaling generates more questions than answers.

Interestingly, time-varying scales are implicit in many software packages and algorithms. To appreciate this point, consider the discretization of a one-dimensional state trajectory, $t \mapsto x \in \mathbb{R}$. For $k = 0, \ldots, N$, the discretized variables x_k represent the samples of the state trajectory; hence, we can write

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} x(t_0) \\ x(t_1) \\ \vdots \\ x(t_N) \end{bmatrix} \tag{53}$$

If the discretized variables are scaled by, say, a diagonal matrix with entries $P_0, \ldots, P_N$, then Eq. (53) transforms according to

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix} \coloneqq \begin{bmatrix} P_0\tilde{x}_0 \\ P_1\tilde{x}_1 \\ \vdots \\ P_N\tilde{x}_N \end{bmatrix} = \begin{bmatrix} x(t_0) \\ x(t_1) \\ \vdots \\ x(t_N) \end{bmatrix} \tag{54}$$

Let $P(t)$ be any function such that $P(t_k) = P_k$, $k = 0, \ldots, N$; then, we can write Eq. (54) as

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix} \coloneqq \begin{bmatrix} P_0\tilde{x}_0 \\ P_1\tilde{x}_1 \\ \vdots \\ P_N\tilde{x}_N \end{bmatrix} \coloneqq \begin{bmatrix} P(t_0)\tilde{x}(t_0) \\ P(t_1)\tilde{x}(t_1) \\ \vdots \\ P(t_N)\tilde{x}(t_N) \end{bmatrix} = \begin{bmatrix} x(t_0) \\ x(t_1) \\ \vdots \\ x(t_N) \end{bmatrix} \tag{55}$$

Hence, it follows that scaling the discretized variables is equivalent to discretizing the continuous-time trajectory according to

$$x(t) = P(t)\tilde{x}(t) \tag{56}$$

Consequently, any algorithm or software package that scales the variables at the discrete level is implicitly using time-varying scales at the optimal-control level. Given this fact, it is critical that the additional dynamics noted in Eq. (52) be automatically incorporated at the discrete level in the algorithm or software package. To understand how this can be done, let Δ_k be any discrete derivative. Because the additional dynamics in Eq. (52) is a consequence of the product rule of continuous calculus, the equivalent discrete product rule,

$$\Delta_k x_k \coloneqq P_k(\Delta_k \tilde{x}_k) + (\Delta_k P_k)\tilde{x}_k, \quad k = 0, \ldots, N \tag{57}$$

must be naturally incorporated in the algorithm or software package. If Δ_k is a forward difference operator, then

$$\Delta_k(P \cdot \tilde{x})_k \coloneqq P_{k+1}\tilde{x}_{k+1} - P_k\tilde{x}_k$$

It is quite straightforward to show that $\Delta_k(P \cdot \tilde{x}) \neq \Delta_k x_k$ and that

$$\Delta_k x_k - \Delta_k(P \cdot \tilde{x})_k = (\Delta_k P_k)(\Delta_k \tilde{x}_k) \tag{58}$$

The right-hand side of Eq. (58) is not necessarily a second-order effect unless the scaling algorithm renders $|(\Delta_k P_k)(\Delta_k \tilde{x}_k)|$ small. Recall that P_k is a scaling factor at the discretized level and not necessarily connected to some continuous function $P(t)$ with a small Lipschitz constant; see Eq. (54). Consequently, if $|(\Delta_k P_k)(\Delta_k \tilde{x}_k)|$ is not small, then the algorithm- any algorithm- is attempting to solve for the wrong dynamics. The implications of this insight are far reaching:

1) If the algorithmic iterations do not converge and/or are expensive (i.e., take a long computational time), it is quite possible the original optimal control problem might have been easy but rendered hard because of scaling at the discretized level.

2) The situation might be made even worse with more sophisticated scaling like adaptive scaling. In such schemes, the scale factors $P_0, P_1, \ldots, P_N$ [see Eq. (55)] are not constants over the course of the iteration but change adaptively based on the current iterate. In following the same process that led to Eq. (56), adaptive scaling implies that in continuous time the state variable is scaled according to some feedback process,

$$x(t) = P(t, x)\tilde{x}(t) \tag{59}$$

over the course of the iterations. Ignoring the possible stability issues resulting from this feedback process, it is clear that this may be worse than time-varying scaling because we now have additional dynamics associated with P,

$$\frac{dP}{dt} = \frac{\partial P}{\partial t} + \frac{\partial P}{\partial x}\frac{dx}{dt} \tag{60}$$

These continuous-time dynamics are not necessarily incorporated in adaptive scaling because the chain rule (of continuous calculus) must also be incorporated at the discrete level (in addition to the product rule).

3) From the preceding point, it is also clear that nonlinear scaling also has the same drawbacks of adaptive scaling (at the discrete level).

4) In an optimistic scenario, is quite possible that $|(\Delta_k P_k)(\Delta_k \tilde{x}_k)|$ is small either by an implicit/explicit result of a scaling algorithm or by accident. Even under this fortuitous case, the error in the satisfaction of the dynamical equations is higher than the computational tolerances

enforced unless of course $|(\Delta_k P_k)(\Delta_k \tilde{x}_k)| = 0$ for $k = 0, \ldots, N$. This is yet another reason why an independent verification of feasibility as highlighted in Sec. VI.C is crucial for validating numerical accuracy.

5) In the best-case scenario, $|(\Delta_k P_k)(\Delta_k \tilde{x}_k)| = 0$ for $k = 0, \ldots, N$. In this case, an algorithm is solving for the correct but transformed dynamics given by Eq. (52). In the absence of new analysis, there is no apparent reason why the transformed dynamics is universally better for optimization than the original dynamics.

The preceding analysis explains why autoscaling done at the discrete level without explicit consideration of the dynamics of the optimal control problem may be harmful to the accuracy and convergence of the algorithm. A simple remedy for this problem is to perform scaling and balancing at the optimal control level and turn off any autoscaling options of software packages that are based on scaling the discretized Jacobian that does not incorporate the additional dynamics presented in Eq. (52).

VIII. Conclusions

Some of the ideas and parts of the process presented in this paper have been used for well over a decade, albeit in an application-specific manner, to generate successful flight implementations. In this paper, we have generalized previous concepts and provided a clear mathematical framework leading to a more unified procedure. More specifically, we have shown that the concept of designer units is fundamentally liberating. If necessary, it even allows one to choose radically different units of measurement along the x and y directions. As a consequence of this liberation, the physical concept of a vector as a quantity with magnitude and direction must be abandoned. A vector is simply a stack of scalar variables in any units. A covector is a measurement conversion device that connects the disparate units of a vector to some common unit. In an optimal control problem, the Lagrange multipliers are covectors, and the common unit of measurement is the cost unit or the cost unit per time unit. The numerical values of a covector have a seesaw effect on the values of the associated vector. The seesaw effect can be used to balance the primal and dual variables for computational efficiency.

It is not necessary to scale and balance the variables on a unit interval; in fact, it may not even be feasible. Even in situations in which the state variables are naturally constrained to the unit interval, it is possible to scale them beyond their physical bounds to achieve better balancing.

Scaling an optimal control problem at the nonlinear-programming level is likely to induce unwanted dynamics that may render an easy problem hard. Therefore, great caution must be exercised in using software packages that simply patch discretization methods to nonlinear programming solvers. In contrast, when scaling and balancing are done at the optimal control level, it can be used quite powerfully to innovate by solving hard problems easily.

Acknowledgments

We thank Steve Paris (formerly of the Boeing Company) for providing the initial spark of using Lagrange multipliers as scaling devices. We also thank the anonymous reviewers for providing many constructive comments, which greatly improved the quality of this paper.

References

[1] Conway, B. A., "A Survey of Methods Available for the Numerical Optimization of Continuous Dynamic Systems," *Journal of Optimization Theory and Applications*, Vol. 152, No. 2, 2012, pp. 271–306. doi:10.1007/s10957-011-9918-z

[2] Conway, B. A., "The Problem of Space Trajectory Optimization," *Space Trajectory Optimization*, edited by B. A. Conway, Vol. 29, Cambridge Aerospace Series, Cambridge Univ. Press, New York, 2010, pp. 1–15.

[3] Longuski, J. M., Guzmán, J. J., and Prussing, J. E., *Optimal Control with Aerospace Applications*, Springer–Verlag, New York, 2014.

[4] Ross, I. M., *A Primer on Pontryagin's Principle in Optimal Control*, 2nd ed., Collegiate, San Francisco, CA, 2015.

[5] Tao, T., *An Epsilon of Room*, Vol. I, American Mathematical Soc., Providence, RI, 2010, Sec. 1.5, pp. 62–75.

[6] McGrath, C. B., Karpenko, M., and Proulx, R. J., "Parallel Generic Algorithms for Optimal Control," *AAS/AIAA Spaceflight Mechanics Conference*, AAS Paper 16-0367, 2016.

[7] McGrath, C. B., Karpenko, M., Proulx, R. J., and Ross, I. M., "Unscented Evolution Strategies for Solving Trajectory Optimization Problems," *AAS/AIAA Spaceflight Mechanics Conference*, AAS Paper 16-0368, 2016.

[8] McGrath, C. B., "Unscented Sampling Techniques for Evolutionary Computation with Applications to Astrodynamic Optimization," Ph.D. Dissertation, Dept. of Mechanical and Aerospace Engineering, Naval Postgraduate School, Sept. 2016.

[9] Bhatt, S., Bedrossian, N., Longacre, K., and Nguyen, L., "Optimal Propellant Maneuver Flight Demonstrations on ISS," *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2013-5027, Aug. 2013.

[10] Kang, W., and Bedrossian, N., "Pseudospectral Optimal Control Theory Makes Debut Flight: Saves NASA $1M in Under 3 Hrs," *SIAM News*, Vol. 40, No. 7, 2007, pp. 1–3.

[11] Bedrossian, N., Bhatt, S., Lammers, M., and Nguyen, L., "Zero Propellant Maneuver: Flight Results for 180° ISS Rotation," *20th International Symposium on Space Flight Dynamics*, NASA CP-2007-214158, Sept. 2007.

[12] Bedrossian, N., Bhatt, S., Kang, W., and Ross, I. M., "Zero Propellant Maneuver Guidance," *IEEE Control Systems Magazine*, Vol. 29, No. 5, 2009, pp. 53–73. doi:10.1109/MCS.2009.934089

[13] Yan, H., Gong, Q., Park, C., Ross, I. M., and D'Souza, C. N., "High Accuracy Trajectory Optimization for a Trans-Earth Lunar Mission," *Journal of Guidance, Control and Dynamics*, Vol. 34, No. 4, 2011, pp. 1219–1227. doi:10.2514/1.49237

[14] Bedrossian, N., Karpenko, M., and Bhatt, S., "Overclock My Satellite: Sophisticated Algorithms Boost Satellite Performance on the Cheap," *IEEE Spectrum Magazine*, Vol. 49, No. 11, 2012, pp. 54–62. doi:10.1109/MSPEC.2012.6341207

[15] Karpenko, M., Ross, I. M., Stoneking, E., Lebsock, K., and Dennehy, C. J., "A Micro-Slew Concept for Precision Pointing of the Kepler Spacecraft," *AAS/AIAA Astrodynamics Specialist Conference*, AAS Paper 15-628, Aug. 2015.

[16] Karpenko, M., Dennehy, C. J., Marsh, H. C., and Gong, Q., "Minimum Power Slews and the James Webb Space Telescope," *27th AAS/AIAA Space Flight Mechanics Meeting*, AAS Paper 17-285, 2017.

[17] Gong, Q., Kang, W., Bedrossian, N., Fahroo, F., Sekhavat, P., and Bollino, K., "Pseudospectral Optimal Control for Military and Industrial Applications," *46th IEEE Conference on Decision and Control*, IEEE Publ., Piscataway, NJ, 2007, pp. 4128–4142.

[18] Karpenko, M., Bhatt, S., Bedrossian, N., and Ross, I. M., "Flight Implementation of Shortest-Time Maneuvers for Imaging Satellites," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 4, 2014, pp. 1069–1079. doi:10.2514/1.62867

[19] Stevens, R. E., and Wiesel, W., "Large Time Scale Optimal Control of an Electrodynamic Tether Satellite," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 6, 2008, pp. 1716–1727. doi:10.2514/1.34897

[20] Ross, I. M., and Karpenko, M., "A Review of Pseudospectral Optimal Control: From Theory to Flight," *Annual Reviews in Control*, Vol. 36, No. 2, 2012, pp. 182–197. doi:10.1016/j.arcontrol.2012.09.002

[21] Minelli, G., Karpenko, M., Ross, I. M., and Newman, J., "Autonomous Operations of Large-Scale Satellite Constellations and Ground Station Networks," *AAS/AIAA Astrodynamics Specialist Conference*, AAS Paper 17-761, 2017.

[22] Karpenko, M., and Proulx, R. J., "Experimental Implementation of Riemann-Stieltjes Optimal Control for Agile Imaging Satellites," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 1, 2016, pp. 144–150. doi:10.2514/1.G001325

[23] Vinter, R. B., *Optimal Control*, Birkhäuser, Boston, MA, 2000.

[24] Clarke, F., *Functional Analysis, Calculus of Variations and Optimal Control*, Springer–Verlag, London, 2013, Chap. 22, pp. 435–472.

[25] Ross, I. M., and Fahroo, F., "Pseudospectral Knotting Methods for Solving Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 3, 2004, pp. 397–405. doi:10.2514/1.3426

[26] Gong, Q., and Ross, I. M., "Autonomous Pseudospectral Knotting Methods for Space Mission Optimization," *AAS Spaceflight Mechanics Meeting*, AAS Paper 06-151, 2006.

[27] Kang, W., Gong, Q., and Ross, I. M., "On the Convergence of Nonlinear Optimal Control Using Pseudospectral Methods for Feedback Linearizable Systems," *International Journal of Robust and Nonlinear Control*, Vol. 17, No. 14, 2007, pp. 1251–1277. doi:10.1002/(ISSN)1099-1239

[28] Gong, Q., Fahroo, F., and Ross, I. M., "Spectral Algorithm for Pseudospectral Methods in Optimal Control," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, 2008, pp. 460–471. doi:10.2514/1.32908

[29] Kang, W., "Rate of Convergence for a Legendre Pseudospectral Optimal Control of Feedback Linearizable Systems," *Journal of Control Theory and Applications*, Vol. 8, No. 4, 2010, pp. 391–405. doi:10.1007/s11768-010-9104-0

[30] Gong, Q., Ross, I. M., and Fahroo, F., "Spectral and Pseudospectral Optimal Control over Arbitrary Grids," *Journal of Optimization Theory and Applications*, Vol. 169, No. 3, 2016, pp. 759–783. doi:10.1007/s10957-016-0909-y

[31] Gong, Q., Kang, W., and Ross, I. M., "A Pseudospectral Method for the Optimal Control of Constrained Feedback Linearizable Systems," *IEEE Transactions on Automatic Control*, Vol. 51, No. 7, July 2006, pp. 1115–1129. doi:10.1109/TAC.2006.878570

[32] Kang, W., Ross, I. M., and Gong, Q., "Pseudospectral Optimal Control and Its Convergence Theorems," *Analysis and Design of Nonlinear Control Systems*, Springer–Verlag, Berlin, 2008, pp. 109–126.

[33] Ross, I. M., "A Roadmap for Optimal Control: The Right Way to Commute," *Annals of the New York Academy of Sciences*, Vol. 1065, No. 1, 2005, pp. 210–231. doi:10.1196/annals.1370.015

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

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

[36] 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.

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

[38] Williams, P., "Jacobi Pseudospectral Method for Solving Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 2, pp. 293–297, 2004.

[39] Fahroo, F., and Ross, I. M., "Pseudospectral Methods for Infinite-Horizon Nonlinear Optimal Control Problems," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, AIAA Paper 2005-6076, 2005.

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

[41] Paris, S. W., Riehl, J. P., and Sjaw, W. K., "Enhanced Procedures for Direct Trajectory Optimization Using Nonlinear Programming and Implicit Integration," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, AIAA Paper 2006-6309, 2006.

[42] Hattis, P. D., "Predictive Momentum Management for the Space Station," *Journal of Guidance, Control, and Dynamics*, Vol. 9, No. 4, 1986, pp. 454–461. doi:10.2514/3.20132

[43] Kyparisis, J., "Sensitivity Analysis for Nonlinear Programs and Variational Inequalities with Nonunique Multipliers," *Mathematics of Operations Research*, Vol. 15, No. 2, 1990, pp. 286–298. doi:10.1287/moor.15.2.286

[44] Wachsmuth, G., "On LICQ and the Uniqueness of Lagrange Multipliers," *Operations Research Letters*, Vol. 41, No. 1, 2013, pp. 78–80. doi:10.1016/j.orl.2012.11.009