Second-Order Switching Time Optimization for Non-linear Time-varying Dynamic Systems

Elliot R. Johnson, *Student Member, IEEE,* Todd D. Murphey, *Member, IEEE,*

Abstract

This paper gives a method for calculating the first and second derivatives of a cost function with respect to switching times for systems with piecewise second-differentiable dynamics. Differential equations governing the linear and bilinear operators required for calculating the derivatives are presented. Example optimizations of linear and non-linear systems are presented as evidence for the value of second-order optimization methods. One example converges in 33 iterations using second-order methods whereas the first-order algorithm requires over 30,000 iterations.

Index Terms

optimization, switched systems

I. Introduction

Switched dynamic systems discontinuously switch from one dynamic function to the next in a known sequence as certain switching times are reached. Each dynamic function (i.e. $\dot{x} = f(x, u, t)$) itself is continuous and differentiable. Switching time optimization is the problem of determining a set of switching times that minimize a cost function.

In this paper the switching times are parameterized as a half space of partially ordered times (i.e., the switching times between two pre-specified modes satisfy an order, so the set of parameterized switching times does not form a vector space). Optimizations are implemented

elliot.r.johnson@u.northwestern.edu

t-murphey@northwestern.edu

Mechanical Engineering, McCormick School of Engineering at Northwestern University, Evanston, IL

This work was supported by the National Science Foundation under grant CCF-0907869.

with standard iterative, numerical algorithms [21], [22] (e.g. the Steepest descent algorithm and Newton's method) that use the local vector space structure. The focus of research has been algorithms to calculate the derivatives of a cost function with respect to the switching times [12], [29], [30], [13], [3], [1], [10], [4], [11], [17], [31]. In [27], the authors convert the switching time problem to a switching time duration problem, thereby converting the optimal control problem to a discrete time optimal control problem that can be solved using constrained algebraic optimization techniques. In general, these works have focused specifically on constrained variational methods by calculating first derivatives [13], [4], [11], [12], [17] using Lagrange multiplier methods, though some have also discussed (but not computed) the second derivative [10], [27], [31]. One exception to this is [7] where the second derivative is found in the case of the underlying space being the space of generalized functions (e.g., differential measures) that generate the switching signals. Another exception is in [16], where the authors discretize in time first to form a nonlinear programming problem, and then optimize the resulting algebraic problem using second-order methods. The downside of such an approach is that time step must be chosen ahead of time rather than allowing one to use adaptive meshes during integration. So, although second-order sensitivity analysis has been studied before, a derivation of its switching time formulation in continuous time has been missing. Lastly, adjoint formulations of parametric sensitivities have been studied in [25] and used in software [28], [18]. The formulas that one obtains for parametric sensitivities are not the same as the ones one obtains for switching times, but they use the same derivative information thereby making software that is appropriate for computing parametric sensitivities (such as CVODES [28], [18]) appropriate for computing derivatives with respect to switching times.

Lastly, it should be mentioned that switching time optimization techniques typically assume that mode order is known, whereas [3], [1], [17], [7], [15] focus on optimizing over mode order as well. Optimizing over mode order has been approached using constrained optimization [3], [1], using the idea of mode "insertions" [12], [15], and using the nonsmooth projection of a smooth optimal control problem [7]. All of these methods focus on generating the number and choice of modes rather than on the efficiency of the calculation. For the present work, we assume that some such technique has already been employed to coarsely estimate the number of modes and that one desires to efficiently find the optimal times that the modes should transition.

This paper—subsets of which can be found in [19], [20]—uses a different approach to

find the same first-order result as [13] that involves fewer steps and relies on fundamental principles in calculus instead of constrained Lagrange multiplier techniques (similar to the previously-mentioned work in [27]). More importantly, the derivation generalizes to find the second derivative of the cost function using an equally simple single integration strategy—a strategy that is much simpler than that found in the generalized function setting in [7]. Given that the difference between linear and quadratic convergence rates are often significant in real-time application [10], [6], [7], having an efficient way of computing adjoint equations for second-order approximations is useful for online optimization. We provide two examples that compare both methods, and a third example that is intractable using first-order techniques.

Section II establishes the precise problem definition. Section III derives the first derivatives of the trajectory and cost function, replicating the result in the prior works cited above using a simpler proof technique. Elements of that proof are used to find the second derivatives of the trajectory and cost in Section IV. Problems that involve terminal costs are discussed in Section V. The resulting optimization is presented in Section VI where optimization is implemented with iterative, numerical algorithms [21], [22] using general purpose software such as MATLAB or *Mathematica* or using software specifically designed for optimal control (such as CVODES [28], [18]). Section VII applies the techniques to three examples that compare first and second-order methods.

II. Problem Definition

Consider an n-dimensional non-linear system governed by a sequence of N dynamic models:

$$\dot{x} = f_i(x,t) \quad \tau_i \leq t < \tau_{i+1}, \tag{1}$$

with $\tau_1 = t_0$ and $\tau_{N+1} = t_f$ defining the time horizon and with initial condition $x(t_0) = x_0$. Each $f_k(x,t)$ is at least C^2 in x and C^1 in t. We seek the $N-1$ switching times[1] $\tau_2 \dots \tau_N$ that optimize a total cost:

$$J(\tau_2, \tau_3 \cdots \tau_N) = \int_{t_0}^{t_f} \ell(x(t),t)\,\mathrm{d}t \tag{2}$$

where $\ell(x,t)$ is an arbitrary C^2 (in x) incremental cost function that is integrable in t. For example we might choose $\ell(x,t) = (x - x_d(t))^T(x - x_d(t))$ to find the switching times that result in

[1]Having τ_2 be the first switching time is awkward, but otherwise the first trajectory will be $x_0(t)$, which is inconsistent with the conventional notion that x_0 is a constant initial condition.

the best possible tracking of the desired trajectory x_d. Terminal cost can also be included, as discussed in Section V. Optimization is approached using iterative numeric algorithms [21], [22] as described in Section VI.

A. Notation

The most important notational point is that we abbreviate a trajectory as $x(t)$ when strictly it should be $x(x_0, \tau_1, \tau_2 \cdots \tau_N, t)$. For any time-dependent function $y(t)$, we refer to a segment $y_k(t)$ to be $y(t) \,\forall\, t \in [\tau_k, \tau_{k+1}]$. Note that by continuity $y_k(\tau_{k+1}) = y_{k+1}(\tau_{k+1})$. For an operator M applied to U we write $M \circ U$. For finite dimensional linear operators, we use square brackets $[M]$ to indicate matrix representations (e.g. $M \circ U = [M]U$ and $M \circ (U, V) = U^T[M]V$).

We use $Df(x)$ notation for derivatives. The slot derivative $D_n f(arg_1, arg_2, ...) \circ (\partial arg_N)$ represents the derivative of $f(\cdot)$ with respect to the n-th argument. Similarly, $D_{var} f(arg_1, arg_2, \dots) \circ (\partial var)$ is the derivative of $f(\cdot)$ with respect to the variable var.

III. First Derivative of $J(\cdot)$

By chain rule, the first derivative of the cost function J in (2) involves the derivative $D_{\tau_i} x(t) \circ \partial\tau_i$ of the trajectory $x(t)$ with respect to each switching time $\tau_i \,\forall\, i = 2 \cdots N$.

Lemma 3.1:

$$D_{\tau_i} x(t) \circ \partial\tau_i = \begin{cases} 0 & t < \tau_i \\ \Phi(t, \tau_i) \circ X^i & t \geq \tau_i \end{cases} \tag{3}$$

$$X^i = \big(f_{i-1}(x(\tau_i), \tau_i) - f_i(x(\tau_i), \tau_i)\big)\partial\tau_i$$

where $\Phi(t, \tau)$ is the state transition matrix [8] for the system $\dot{x} = [D_1 f(x(t), t)]x$.
The proof of this is a direct application of chain rule to the integral representation of $x(t)$ (and can be found in [19]). Lemma (3) allows us to compute the derivative of the cost with respect to switching time τ_i using chain rule.

Lemma 3.2: The derivative of the cost in Eq. (2) with respect to each switching time τ_i is

$$D_{\tau_i} J(\cdot) \circ \partial\tau_i = \psi(t_f, \tau_i) \circ X^i \tag{4}$$

where $\psi(t_f, \tau) : \mathbb{R}^n \to \mathbb{R}$ is found by integrating

$$\psi(t_f, t_f) \circ U = 0 \tag{5a}$$

$$\tfrac{\partial}{\partial\tau}\psi(t_f, \tau) \circ U = -D_1 \ell(x(\tau), \tau) \circ U - \psi(t, \tau) \circ D_1 f(x(\tau), \tau) \circ U \tag{5b}$$

backwards along τ from t_f to τ_i.

Proof: Take the derivative of Eq. (2) with respect to τ_i. The resulting integrand from t_0 to τ_i is zero because $D_{\tau_i}x(t) = 0$ for $t < \tau_i$, leaving

$$D_{\tau_i}J(\cdot) \circ \partial\tau_i = \int_{\tau_i}^{t_f} D_1\ell(x(s),s) \circ D_{\tau_i}x(s) \circ \partial\tau_i \,\mathrm{d}s = \underbrace{\int_{\tau_i}^{t_f} D_1\ell(x(s),s) \circ \Phi(s,\tau_i)\,\mathrm{d}s}_{\psi(t_f,s)} \circ X^i. \quad (6)$$

Differentiating $\psi(t_f,\tau)$ with respect to τ and integrating backwards from t_f to τ_2 will find all the values of $\psi(t_f,\tau_i)$ in a single integration. To see this, evaluate $\psi(t_f,\tau_i)$ with $\tau = t$ to find the initial condition (5a) for the integration and differentiate $\psi(t_f,\tau_i)$ with respect to τ:

$$\begin{aligned}\tfrac{\partial}{\partial\tau}\psi(t_f,\tau) \circ U &= -D_1\ell(x(\tau),\tau) \circ \Phi(\tau,\tau) \circ U - \int_{\tau}^{t} D_1\ell(x(s),s) \circ \Phi(s,\tau) \circ D_1f(x(\tau),\tau) \circ U \,\mathrm{d}s \\ &= -D_1\ell(x(\tau),\tau) \circ U - \psi(t,\tau) \circ D_1f(x(\tau),\tau) \circ U \quad (7)\end{aligned}$$

where the first term comes from the Leibniz integral rule and we have used state transition matrix identities [8]. This proves the final statement of the Lemma. ■

This result has previously been reported by [13], [31], where it was derived using constrained variational methods that rely upon Lagrange multipliers (that can have problems with abnormal extrema [14]). Here we have only used chain rule. Importantly, this approach extends to the second derivative.

IV. Second Derivative of $J(\cdot)$

The second derivative of the cost is found using the same strategy from Sec. III. We find a differential equation describing the second derivative and then show that the solutions can be expressed with a state transition matrix and a bilinear operator.

The second derivative of $x(t)$ is symmetric (i.e, mixed partials commute), so we assume $i \geq j$ for the remainder of the paper without loss of generality. This is only for brevity; the same strategy found below will find the full second derivative without assuming symmetry a priori.

Proposition 4.1: With $i \geq j$ (and $t \geq \tau_i$), the second derivative of the trajectory satisfies a differential equation (8a) with initial condition (8b).

$$\frac{d}{dt}D_{\tau_j}D_{\tau_i}x(t)\circ(\partial\tau_j,\partial\tau_i) = D_1f(x(t),t)\circ D_{\tau_j}D_{\tau_i}x(t)\circ(\partial\tau_j,\partial\tau_i) + D_1^2f(x(t),t)\circ(D_{\tau_j}x(t)\circ\partial\tau_j, D_{\tau_i}x(t)\circ\partial\tau_i) \tag{8a}$$

$$D_{\tau_j}D_{\tau_i}x(\tau_i)\circ(\partial\tau_j,\partial\tau_i) = \begin{cases} D_1f_i(x(\tau_i),\tau_i)\circ f_i(x(\tau_i),\tau_i)\partial\tau_j\partial\tau_i \\ \quad + D_1f_{i-1}(x(\tau_i),\tau_i)\circ f_{i-1}(x(\tau_i),\tau_i)\partial\tau_j\partial\tau_i \\ \quad - 2D_1f_i(x(\tau_i),\tau_i)\circ f_{i-1}(x(\tau_i),\tau_i)\partial\tau_j\partial\tau_i \\ \quad + D_2f_{i-1}(x(\tau_i),\tau_i)\circ\partial\tau_j\partial\tau_i - D_2f_i(x(\tau_i),\tau_i)\circ\partial\tau_j\partial\tau_i & i=j \\ \big(D_1f_{i-1}(x(\tau_i),\tau_i) - D_1f_i(x(\tau_i),\tau_i)\big)\circ\Phi(\tau_i,\tau_j)\circ X^j\partial\tau_i & i>j \end{cases} \tag{8b}$$

The proof of this result is again based directly on chain rule and can be found in [19]. Unlike the first derivative, the ODE for the second derivative (8a) is not linear, but it is affine. We can use the linear model to express the second derivative in terms of a state transition matrix and a new bilinear operator, $\phi(t,\tau)$ that is analogous to $\Phi(t,\tau)$ in Lemma 3.1.

Lemma 4.2: The second derivative $D_{\tau_j}D_{\tau_i}x(t)\circ(\partial\tau_j,\partial\tau_i)$ is

$$D_{\tau_j}D_{\tau_i}x(t)\circ(\partial\tau_j,\partial\tau_i) = \Phi(t,\tau_i)\circ X^{i,j} + \phi(t,\tau_i)\circ\big(\Phi(\tau_i,\tau_j)\circ X^j, X^i\big) \tag{9}$$

where $\Phi(t,\tau)$ is the state transition matrix in Lemma 3.1 and $\phi(t,\tau):\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}^n$ is the bilinear operator that satisfies

$$\phi(t,t)\circ(U,V) = 0 \tag{10a}$$

$$\frac{\partial}{\partial\tau}\phi(t,\tau)\circ(U,V) = -\Phi(t,\tau)\circ D_1^2f(x(\tau),\tau)\circ(U,V) - \phi(t,\tau)\circ\big(D_1f(x(\tau),\tau)\circ U,V\big) - \phi(t,\tau)\circ\big(U,D_1f(x(\tau),\tau)\circ V\big). \tag{10b}$$

and $X^{i,j}$ is the initial condition from (8b).

(The proof of this result is not critical to the presentation here, so the reader may find it in [20].) Equations (10) facilitate generalization of Sec. III to the second derivative of the cost.

Theorem 4.3: The second derivative of $J(\cdot)$ with respect to switching times τ_j and $\tau_i\geq\tau_j$ is

$$D_{\tau_j}D_{\tau_i}J(\cdot)\circ(\partial\tau_j,\partial\tau_i) = -D_1\ell(x(\tau_i),\tau_i)\circ X^i\partial\tau_j\delta_i^j + \psi(t_f,\tau_i)\circ X^{i,j} + \Omega(t_f,\tau_i)\circ(\Phi(\tau_i,\tau_j)\circ X^j, X^i) \tag{11}$$

where δ_i^j is the Kronecker delta and $\Omega(t,\tau)\circ(U,V) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is the bilinear operator found by integrating

$$\Omega(t,t)\circ(U,V) = 0_{n\times n} \tag{12a}$$

$$\begin{aligned}\tfrac{\partial}{\partial\tau}\Omega(t,\tau)\circ(U,V) = &-D_1^2\ell(x(\tau),\tau)\circ(U,V) - \psi(t,\tau)\circ D_1^2 f(x(\tau),\tau)\circ(U,V) - \\ &\Omega(t,\tau)\circ\big(D_1 f(x(\tau),\tau)\circ U, V\big) - \Omega(t,\tau)\circ\big(U, D_1 f(x(\tau),\tau)\circ V\big)\end{aligned} \tag{12b}$$

backwards over τ from t_f to τ_i.

Proof: Take the derivative of (6) with respect to τ_j:

$$\begin{aligned}D_{\tau_j}D_{\tau_i}J(\cdot)\circ(\partial\tau_j,\partial\tau_i) &= \tfrac{\partial}{\partial\tau_j}\bigg(\int_{\tau_i}^{t_f} D_1\ell(x(s),s)\circ D_{\tau_i}x(s)\circ\partial\tau_i\,\mathrm{d}s\bigg) \\ &= -D_1\ell(x(\tau_i),\tau_i)\circ D_{\tau_i}x(\tau_i)\circ\partial\tau_i\partial\tau_j\delta_j^i + \int_{\tau_i}^{t_f} D_1\ell(x(s),s)\circ\Phi(s,\tau_i)\circ X^{i,j}\,\mathrm{d}s \\ &\quad + \int_{\tau_i}^{t_f} D_1\ell(x(s),s)\circ\phi(s,\tau_i)\circ\big(\Phi(\tau_i,\tau_j)\circ X^j, X^i\big) \\ &\qquad + D_1^2\ell(x(s),s)\circ\big(\Phi(s,\tau_i)\circ\Phi(\tau_i,\tau_j)\circ X^j, \Phi(s,\tau_i)\circ X^i\big)\,\mathrm{d}s \\ &= -D_1\ell(x(\tau_i),\tau_i)\circ D_{\tau_i}x(\tau_i)\circ\partial\tau_i\partial\tau_j\delta_j^i + \psi(t_f,\tau_i)\circ X^{i,j} + \Omega(t_f,\tau_i)\circ(\Phi(\tau_i,\tau_j)\circ X^j, X^i)\end{aligned}$$

where $\Omega(t,\tau)$ is defined as

$$\Omega(t,\tau)\circ(U,V) = \int_\tau^t D_1\ell(x(s),s)\circ\phi(s,\tau)\circ(U,V) + D_1^2\ell(x(s),s)\circ(\Phi(s,\tau)\circ U, \Phi(s,\tau)\circ V)\,\mathrm{d}s. \tag{13}$$

Equations (13) and (10) provide a way to calculate $D_{\tau_j}D_{\tau_i}J(\cdot)$ by forward integration. In doing so, we must integrate (13) to find $\Omega(t_f,\tau_i)$ for each τ_i. By integrating backwards along τ, we reduce this to a single integration. The initial condition for the backwards integration is seen to be zero directly from the definition of $\Omega(t,\tau)$, given in (12a) of the Lemma. Equation (12b) of the Lemma is found by differentiating (13) with respect to τ and applying identities for $\Phi(t,\tau)$ [8] and the identities for $\psi(t,\tau)$ found in (10). ■

Theorem 4.3 is the natural extension of Lemma 3.2 to the second derivative. It has the property that $\phi(t,\tau)$, the second-order analog to the state transition matrix, is no longer required and provides an algorithm for calculating every second derivative from a single integration. It is useful to write (12b) in matrix form to see how it is calculated as a matrix-valued ordinary

differential equation:

$$\begin{aligned}\tfrac{\partial}{\partial\tau}[\Omega(t,\tau)] = -[D_1^2\ell(x(\tau),\tau)] - [\psi(t,\tau)\circ D_1^2 f(x(\tau),\tau)] \\ - [D_1 f(x(\tau),\tau)]^T[\Omega(t,\tau)] - [\Omega(t,\tau)][D_1 f(x(\tau),\tau)].\end{aligned}$$

While Theorem 4.3 avoids $\phi(t,\tau)$, it does rely on $\Phi(t,\tau)$. The state transition matrix for the first derivative has re-appeared in the initial condition for the second derivative (8b) and the second derivative of the cost in Eq. (11).

V. Including a Terminal Cost

The integral cost function in (2) is often augmented with a terminal cost $m(x,t)$, here assumed to be C^2 in x:

$$J(\cdot) = \int_{t_0}^{t_f} \ell(x(t),t)\,\mathrm{d}t + m(x(t_f),t_f). \tag{14}$$

For example, if we care more about the final state of the system than how it gets there, the terminal cost can be weighted heavily compared to the incremental cost.

Lemma 5.1: Lemma 3.2 and Theorem 4.3 hold for the additional terminal cost in (14) if the initial conditions for (5a) and (12a) are changed to

$$\psi(t,t)\circ U = D_1 m(x(t),t)\circ U \qquad \Omega(t,t)\circ(U,V) = D_1^2 m(x(t),t)\circ(U,V)$$

Proof: The proof proceeds by taking the derivatives of (14) and following the proofs for the first and second derivatives of the original cost (Lemma 3.2 and Theorem 4.3). In the derivations, $\psi(t,\tau)$ becomes

$$\psi(t,\tau)\circ U = D_1 m(x(t),t)\circ\Phi(t,\tau)\circ U + \int_\tau^t D_1\ell(x(s),s)\circ\Phi(s,\tau)\circ U\,\mathrm{d}s$$

and $\Omega(t,\tau)$ becomes

$$\begin{aligned}\Omega(t,\tau)\circ(U,V) = D_1 m(x(t),t)\circ\phi(t,\tau)\circ(U,V) + D_1^2 m(x(t),t)\circ(\Phi(t,\tau)\circ U,\Phi(t,\tau)\circ V) \\ + \int_\tau^t D_1\ell(x(s),s)\circ\psi(s,\tau)\circ(U,V) + D_1^2\ell(x(s),s)\circ(\Phi(s,\tau)\circ U,\Phi(s,\tau)\circ V)\,\mathrm{d}s.\end{aligned}$$

The new operators have different initial conditions, but we find their differential equations along τ are unchanged. The rest of the proof proceeds exactly as the case without terminal cost. ■

VI. Optimization Algorithm

We optimize Eq. (2) with iterative approaches [22] that rely on first and second derivatives. In each iteration, we choose a descent direction $z = -[H]^{-1}[D_\tau J(\cdot)]^T$ where H is a positive semidefinite matrix. Choosing $H = I$ gives the steepest descent algorithm—yielding a first-order optimization that has linear convergence. Choosing $H = D_\tau D_\tau J(\cdot)$ results in Newton's method, a second-order optimization with local quadratic convergence. When the Hessian is not positive definite, there are several common approaches. The simplest is to revert to the steepest descent direction. Another [22], [24] is to decompose the Hessian into its eigenvalues λ and corresponding eigenvectors P. Negative or near-zero eigenvalues are replaced with 1 and the modified Hessian is reconstructed as $H = P\lambda P^{-1}$. This method–which we will reference as a *quasi-Newton* method–essentially uses Newton's Method in each eigenvector direction with a positive eigenvalue and steepest descent in directions with negative eigenvalues. In general, one should use trust region methods [24] to perform the optimizations with superlinear convergence, but we use the algorithm just outlined partially because it highlights the transition from linear convergence to quadratic convergence. Both algorithms benefit from the *Armijo Line Search* [2] algorithm. This algorithm reduces the magnitude of the step size until there is a sufficient decrease. Satisfying the sufficient decrease condition guarantees that the optimization will eventually converge.

We must also keep the switching times ordered properly (i.e. $\tau_{k+1} \geq t_k$). In this work, after calculating the descent direction z, we find the largest $\epsilon \in (0, 1]$ such that $x + \epsilon z$ is ordered properly. (Note that we choose $\epsilon \neq 0$ because z is a descent direction, so there exists an ϵ_0 small enough such that $J(x + \epsilon z) < J(x)$ for all $\epsilon < \epsilon_0$.)

It is worth mentioning that the computational complexity of this algorithm is constant with respect to the number of switching times, but goes up with the number states N. Computation of the gradient requires solving N differential equations, while computation of the Hessian requires solving $\frac{1}{2}N(N+1)$ differential equations (because of the symmetry of the Hessian). As the next section illustrates, this increase in complexity is often offset by substantial improvements in convergence.

VII. Examples

We present three examples that show the value of a second-order algorithm in switching time optimization. The first example is the same linear time-invariant (LTI) optimization described in

[13]. The second is an optimization of a non-linear system. The third uses Lie brackets [23] to design a sequence of inputs and optimizes the switching times to move to a final configuration.

A. Optimizing an LTI System

This example compares the optimization of a linear time-invariant system using first-order and second-order algorithms. This example problem comes from [13], where it was optimized with a first-order algorithm. The cost function is $\ell(x,t) = \frac{1}{2}x^T x$ and the system is

$$A_1 = \begin{bmatrix} -1 & 0 \\ 1 & 2 \end{bmatrix} \qquad A_2 = \begin{bmatrix} 1 & 1 \\ 1 & -2 \end{bmatrix} \qquad \tfrac{d}{dt}x = f(x,t) = \begin{cases} A_1 x & 0 \le t < \tau_2 \\ A_2 x & \tau_2 \le t < \tau_3 \\ A_1 x & \tau_3 \le t < \tau_4 \\ A_2 x & \tau_4 \le t < 1, \end{cases}$$

with initial condition $x_0 = [1,0]^T$. For the optimization, we choose the Armijo parameters to be $\alpha = 0.2$ and $\beta = 0.7$. The initial switching times are $\tau_2 = 0.3$, $\tau_3 = 0.5$, and $\tau_4 = 0.7$. The terminating condition is $|D_\tau J(\tau_2, \tau_3, \tau_4)| < 10^{-4}$. The second-order optimization takes 10 initial first-order steps.

The first-order optimization converged to $\tau_2 = 0.518$, $\tau_3 = 0.688$, and $\tau_4 = 0.791$ in 302 steps. The second-order optimization converge to the same switching times in 18 steps. Logarithmic plots of the norm of the cost derivative are shown in Fig. 1. Even for this LTI system, the second-order descent reduces the number of iterations by an order of magnitude.

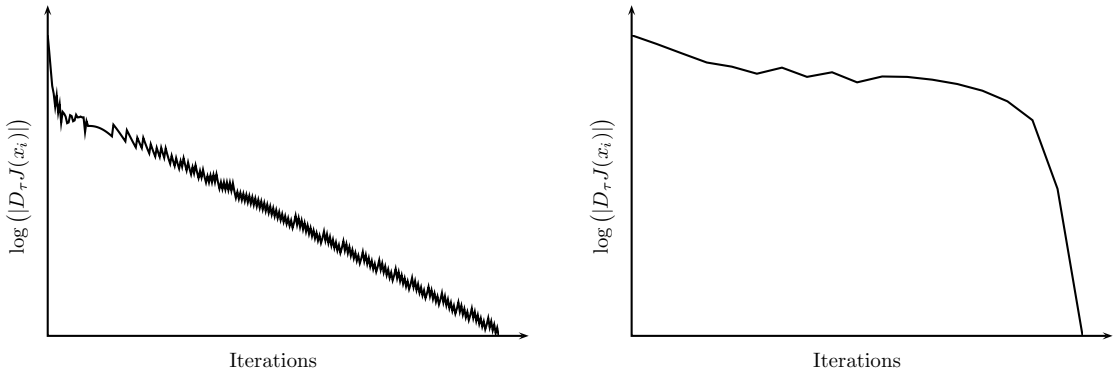
B. Kinematic Car

Our second example compares first and second-order optimizations for a non-linear system. We consider the kinematic car with the equations of motion:

$$\dot{x} = [\dot{X}, \dot{Y}, \dot{\theta}, \dot{\phi}]^T = f_{car}(x,u) = [u_1 \cos(\theta), u_1 \sin(\theta), u_1 \tan(\phi), u_2]^T. \tag{15}$$

The kinematic car can be modeled as a switched system when the control inputs are piecewise-constant: $f_i(x) = f_{car}(x, (u_1, u_2)_i)$. For this example, we consider a parallel parking maneuver made up of 7 sequential inputs:

$$\{(u_1, u_2)\}_{i=0}^{6} = [(0.3, 0), (0, -2.8), (-0.2, 0), (0, 2.9), (-0.09, 0), (0, -1.8), (0.09, 0)].$$



(a) First-Order Optimization

(b) Second-Order Optimization

Fig. 1: Logarithmic plots of the norm of the cost derivative over each iteration of the optimization, with both plots having maximum log magnitudes of 10^0 and minimum log magnitudes of 10^{-4}. The first-order plot (of 302 iterations) shows the expected linear shape, while the second-order plot (of 18 iterations) shows the characteristic quadratically decreasing shape.

For the optimization, we consider the magnitude and sequence of the inputs to be known but the switching times are unknown. The incremental cost function is $\ell(x,t) = (x-x_d)^T(x-x_d)$ and zero terminal cost. The optimization parameters are $\alpha = 0.6$, $\beta = 0.0001$, and the termination condition is $|D_\tau J| < 10^{-4}$. The initial switching times were equal intervals from 0 to 7 seconds (i.e., $\tau_i = i - 1$). Both optimizations converged to the correct switching times that generated the reference trajectory x_d: $(\tau_1 \cdots \tau_{N+1}) = (0,\ 1,\ 1.5,\ 2.5,\ 3.5,\ 4.414,\ 5.248,\ 7)$. The first-order optimization took over 30,000 steps to converge. The second-order optimization, which initially took 10 first-order steps, converges in 24 iterations. Fig. 2 shows the trajectories for each iteration of the optimization.

It is likely that 30,000 iterations is impractical for applications, particularly real-time ones [10]. The second-order optimization reduces this by three orders of magnitude. The next example will use the same system for a more ambitious application of switching time optimization.

C. Kinematic Car: Lie Bracket Trajectory

Lie brackets [5], [9], [26] are infinitesimal operations that take advantage of two vector field not commuting to locally produce motion in a direction outside the linear span of the vector

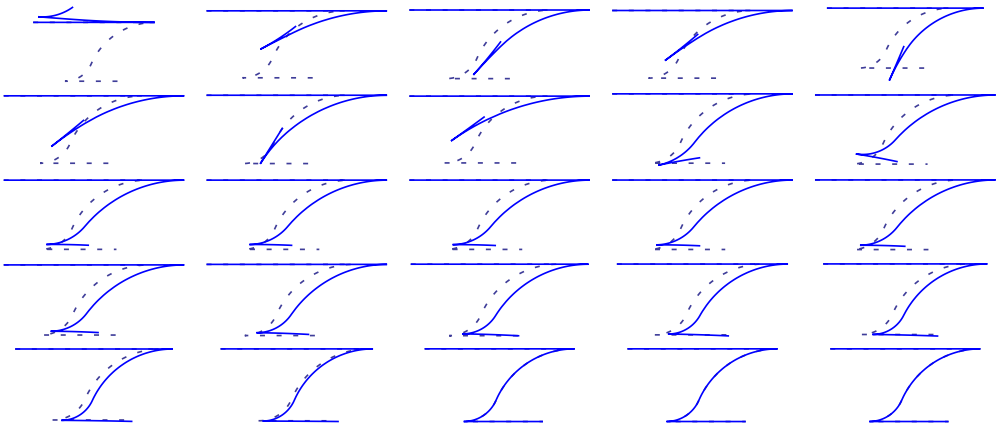


Fig. 2: Trajectories for the kinematic car at each iteration of the optimization. The solid line is the kinematic car and the dashed line is the desired trajectory.

fields. The above parallel parking maneuver is a common example of Lie bracket motion. The Lie bracket is formally based on infinitesimal motion, but it can be used to determine a sequence of inputs for motion planning.

Suppose we seek a trajectory from $q = (0, 0, 0, 0)$ to $q = (0, -1, 0, 0)$. The Lie bracket to produce sideways motion for the kinematic car (15) is the nested bracket

$$\left[\frac{\partial f_{car}}{\partial u_1}, \left[\frac{\partial f_{car}}{\partial u_2}, \frac{\partial f_{car}}{\partial u_1}\right]\right] = (0, -1, 0, 0) \tag{16}$$

which corresponds to infinitesimal movements with the following sequence of inputs:

$$\{(u_1, u_2)\}_{i=0}^{9} = [(1,0), (0,1), (1,0), (0,-1), (-1,0), (-1,0), (1,0), (0,1), (-1,0), (0,-1)].$$

We set up an optimization using the above inputs[2] and add a final "constant" function (i.e. $u = [0, 0]$) to give the optimization flexibility in the duration of the maneuver. Since any trajectory is acceptable, we choose a zero incremental cost and non-zero terminal cost: $\ell(x, t) = 0$ and $m(x, t) = ||x - (0, -1, 0, 0)||^2$. The optimization was run with $\alpha = 0.001$, $\beta = 0.6$, $|D_\tau J| < 10^{-4}$ and initial condition[3] $(\tau_1 \cdots \tau_{N+1}) = (0,\ 2,\ 2.5,\ 3,\ 3.5,\ 5.5,\ 6,\ 6.5,\ 7,\ 12.5)$.

[2]Note that three consecutive movements $[1, 0]$, $[1, 0]$, and $[-1, 0]$ were collapsed into a single movement.

[3]The Lie bracket also suggests the initial (relative) timing: the two fields in each bracket should be about the same length. For example, consider the Lie bracket $[f, [g, h]]$. If we move along f for ϵ seconds, the $[g, h]$ movement should also be ϵ seconds.

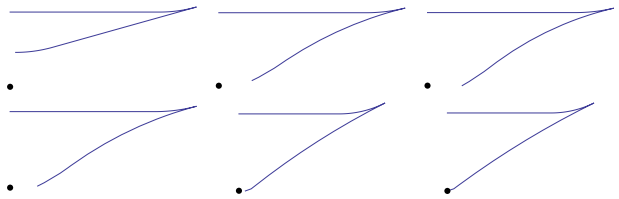


Fig. 3: The trajectory at the 1st, 5th, 10th, 15th, 20th, and 22nd (final) iteration. The dot shows the desired final position.

The optimization took 10 steepest descent steps followed by 12 quasi-Newton steps for a total of 22 iterations. The steepest descent optimization did not converge after 30,000 iterations. Figure 3 shows the final trajectory with several intermediate trajectories. The final switching times were $(\tau_1 \cdots \tau_{N+1}) = (0,\ 1.33,\ 2.01,\ 2.58,\ 3.34,\ 5.45,\ 6.97,\ 7.06,\ 8.50,\ 12.5)$.

To generalize to arbitrary end points in a neighborhood of the initial condition, one need only change the terminal condition of the optimization (and potentially compute the infinitesimal approximation of an end point using the Lie bracket, typically using the Campbell-Baker-Hausdorff expansion [26]. If the distance to be traversed is larger than what a single motion can provide, then computing multiple Lie bracket motions—and therefore optimizing over more switching times—may be desirable.

VIII. Conclusions and Future Work

We have presented derivations for the first and second derivatives of the cost function with respect to switching times. Both derivatives have efficient operator-based representations that do not rely on constrained optimization techniques. The derivation is direct, uses basic calculus tools, and generalizes to second derivatives. Examples demonstrate the expected quadratic convergence. Future work includes considering combined trajectory and switching time optimization where a set of switching times and time-varying inputs minimize a cost function.

References

[1] M. Alamir and S. A. Attia. On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms. *In Proceedings of 6th IFAC Symposium on Nonlinear Control Systems*, pages 558–563, 2004.

[2] L. Armijo. Minimization of functions having lipschitz continuous first-partial derivatives. *Pacific Journal of Mathematics*, Vol. 16, 1966.

[3] S. A. Attia, M. Alamir, and C. Canudas de Wit. Sub optimal control of switched nonlinear systems under location and switching constraints. *In Proceedings of the 16th IFAC World Congress*, 2005.

[4] H. Axelsson, Y. Wardi, M. Egerstedt, and E.I. Verriest. Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. *Journal of Optimization Theory and Applications*, 136:167–186, 2008.

[5] F. Bullo and A.D. Lewis. *Geometric Control of Mechanical Systems*. Number 49 in Texts in Applied Mathematics. Springer-Verlag, 2004.

[6] T. Caldwell and T. D. Murphey. Second-order optimal estimation of slip state for a simple slip-steered vehicle. In *IEEE Int. Conf. on Automation Science and Engineering (CASE)*, pages 133–139, 2009.

[7] T. Caldwell and T. D. Murphey. Switching mode generation and optimal estimation with application to skid-steering. *Automatica*, 47(1):50–64, 2011.

[8] C.T. Chen. *Linear System Theory and Design*. Saunders College Publishing, 1984.

[9] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion*. The MIT Press, 2005.

[10] X.C. Ding, Y. Wardi, and M. Egerstedt. On-line optimization of switched-mode dynamical systems. *IEEE Transactions on Automatic Control*, 54(9):2266–2271, 2009.

[11] M. Egerstedt, S. Azuma, and Y. Wardi. Optimal timing control of switched linear systems based on partial information. *Nonlinear Analysis: Theory, Methods and Applications*, 65:1736–1750, 2006.

[12] M. Egerstedt, Y. Wardi, and H. Axelsson. Transition-time optimization for switched-mode dynamical systems. *IEEE Transactions on Automatic Control*, 51(1):110–115, 2006.

[13] M. Egerstedt, Y. Wardi, and F. Delmotte. Optimal control of switching times in switched dynamical systems. *IEEE Conference on Decision and Control*, pages 2138 – 2143, 2003.

[14] C. A. Floudas and P. M. Pardalos, editors. *Encyclopedia of Optimization*. Springer, 2008.

[15] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. Sastry, R. Bajcsy, and C. Tomlin. A descent algorithm for the optimal control of constrained switched nonlinear dynamical systems. In *Hybrid Systems: Computation and Control*, 2010.

[16] R. Hannemann and W. Marquardt. Continuous and discrete composite adjoints for the hessian of the lagrangian in shooting algorithms for dynamic optimization. *SIAM Journal of Scientific Computing*, 31:4675–4695, 2010.

[17] S. Hedlund and A. Rantzer. Convex dynamic programming for hybrid systems. *IEEE Transactions on Automatic Control*, 47:1536–1540, 2002.

[18] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. Sundials: Suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.*, 31:363–396, 2005.

[19] E. Johnson and T. D. Murphey. Second order switching time optimization for time-varying nonlinear systems. In *IEEE Int. Conf. on Decision and Control (CDC)*, pages 5281 – 5286, 2009.

[20] E. Johnson and T. D. Murphey. Local planning using switching time optimization. In *IEEE Int. Conf. on Automation Science and Engineering (CASE)*, pages 828–834, 2010.

[21] C.T. Kelly. *Iterative Methods of Linear and Nonlinear Equations*. SIAM, 1995.

[22] C.T. Kelly. *Iterative Methods for Optimization*. SIAM, 1999.

[23] R.M. Murray, Z. Li, and S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[24] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2000.

[25] D. B. Özyurt and P. I. Barton. Cheap second order directional derivatives of stiff ode embedded functionals. *SIAM J. Sci. Comput.*, 26(5):1725–1743, 2005.

[26] S. S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Springer-Verlag, New York, 1999.

[27] A. Schild, X. C. Ding, M. Egerstedt, and J. Lunze. Design of optimal switching surfaces for switched autonomous systems. In *IEEE Int. Conf. on Decision and Control (CDC)*, pages 5293–5298, 2009.

[28] R. Serban and A. C. Hindmarsh. CVODES: An ode solver with sensitivity analysis capabilities. *ACM Transactions on Mathematical Software, Proceedings of ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, pages 1–13, 2003.

[29] X. Xu and P.J. Antsaklis. Optimal control of switched autonomous systems. In *IEEE Conference on Decision and Control*, pages 4401 – 4406, 2002.

[30] X. Xu and P.J. Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):2– 16, 2004.

[31] Xuping Xu and Panos J. Antsaklis. Optimal control of switched systems via non-linear optimization based on direct differentiations of value functions. *International Journal of Control*, 75:1406 – 1426, 2002.