Gradient Descent Approach to Optimal Mode Scheduling in Hybrid Dynamical Systems[1]

H. AXELSSON,[2] Y. WARDI,[3] M. EGERSTEDT,[4] and E.I. VERRIEST[5]

Communicated by E. Polak

Abstract. This paper concerns the problem of optimally scheduling the sequence of dynamic response functions in nonlinear switched-mode hybrid dynamical systems. The control parameter has a discrete component and a continuous component, namely the sequence of modes and the duration of each mode, while the performance criterion consists of a cost functional on the state trajectory. The problem is naturally cast in the framework of optimal control. This framework has established techniques sufficient to address the continuous part of the parameter, but lacks adequate tools to consider the discrete element. To get around this difficulty, the paper proposes a bilevel hierarchical algorithm. At the lower level, the algorithm considers a fixed mode sequence and minimizes the cost functional with respect to the modes durations; at the upper level, it updates the mode-sequence by using a gradient technique that is tailored to the special structure of the discrete variable (mode sequencing). The resulting algorithm is not defined on a single parameter space, but rather on a sequence of Euclidean spaces of increasing dimensions, an unusual setting for which there is no established notion of convergence. The paper suggests first a suitable definition of convergence based on the concepts of optimality functions; then, it proves that the proposed algorithm converges in that sense.

Key Words. Switched-mode systems, gradient descent, optimality functions, optimal control.

1. Introduction

Consider a dynamical system that switches among various modes during the course of its operation. Given a cost functional defined on the state trajectory of the system, the problem addressed in this paper is how to schedule the modes in order to minimize the cost functional. Such optimal scheduling problems arise in a number of application domains, including situations where a control module has to switch its attention among a number of subsystems (Refs. 1-3) or collect data sequentially from a number of sensory sources (Refs. 4-6). The underlying system is assumed to be generally nonlinear and the paper focuses on an algorithmic approach, especially on asymptotic convergence.

The scheduling variable is comprised of two parameters: one discrete, and the other continuous. The discrete parameter consists of the sequence of modes, while the continuous parameter consists of the amount of time each mode is in effect. We refer to these parameters as the *sequencing variable* and the *timing variable*, respectively. From the standpoint of optimization, it is generally much easier to deal with the timing variable than with the sequencing variable. After all, for a given sequence of modes, the optimal timing problem amounts to a nonlinear programming problem which can be solved by standard techniques. On the other hand, the question of the optimal sequencing concerns a discrete variable and it may have an exponential complexity. As a matter of fact, several algorithms for the timing problem (with given fix sequencing), tailored to the special structure of switched-mode systems, have been analyzed in Refs. 7-10, while systematic approaches to the sequencing problem,

[1]Research supported in part by the National Science Foundation under Grant #0509064.

[2]Senior Software Engineer, Kencast, Stamford, Conneticut.

[3]Professor, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia.

[4]Associate Professor, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia.

[5]Professor, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia.

largely based on the maximum principle (Refs. 11-13, and 7-8), have just begun to emerge (Refs. 14-15).

Having the scheduling-optimization problem in mind, Ref. 16 has illustrated a way to improve on a given mode sequence by inserting a new mode over a brief period of time in such a way that the cost functional is reduced. This has motivated the development of an algorithm that alternates between two phases: in the first phase, it solves the timing optimization problem for a given mode sequence; in the second phase, it modifies the sequence by inserting to it a new mode at a suitable time. In the latter phase, the algorithm uses local sensitivity information; hence, it is essentially a local-search algorithm. Thus, it seeks only a local minimum to the scheduling problem (in a sense defined below); thus, it appears to evade the complexity issue that is inherent in the search for globally optimal schedules.

In the first phase, the algorithm solves a constrained nonlinear-programming problem whose dimension is related to the number of modes. This number tends to grow since a new mode is inserted each time the algorithm enters this phase. Thus, the algorithm can be viewed as operating not on a single parameter space, but rather on a set of nested Euclidean spaces having increasing dimensions. This framework is unusual and it raises fundamental questions about asymptotic convergence of the algorithm. Although some numerical results have been presented (see Ref. 17), no comprehensive convergence analysis has been developed as of now; however, see Ref. 18 for initial results.

The issue of convergence raises two questions: (i) What is a proper meaning of convergence of such an algorithm? (ii) Does the specific algorithm in question converge in that sense? The present paper addresses these two questions; as we shall see, the answer (especially to the second question) is by-no-means trivial.

The rest of the paper is organized as follows. Section 2 establishes an abstract setting for the optimization problem and defines a suitable concept of algorithms convergence. Section 3 formulates the optimal mode-scheduling problem and recalls some preliminary results. Section 4 presents a convergent algorithm and Section 5 contains an example. Finally, Section 6 concludes the paper.

2. Conceptual Framework for Algorithms Convergence

Consider the problem of minimizing a function $J : \Xi \to R$, where Ξ is a set. Let Γ denote an optimality condition, and let $\Sigma \subset \Xi$ denote the set of points $\xi \in \Xi$ where Γ is satisfied. Let $\theta : \Xi \to R^-$ be a nonpositive-valued function such that $\theta(\xi) = 0$ if and only if $\xi \in \Sigma$.[6] Such a function is called *optimality function associated with* Γ (see Ref. 19, p.19). Typically θ is defined so as to provide a quantitative measure of the extent to which a point $\xi \in \Xi$ satisfies the condition Γ. Its upper semicontinuity in the case where Ξ is a topological space ensures that if $\hat{\xi}$ is an accumulation point of a sequence $\{\xi_j\}_{j=1}^{\infty}$ computed by an algorithm, and if $\lim_{j\to\infty} \theta(\xi_j) = 0$, then $\theta(\hat{\xi}) = 0$ and hence $\hat{\xi} \in \Sigma$. Ref. 19 has used the concept of optimality functions to develop a unified approach to analysis and design of optimization algorithms, providing simple proofs of their asymptotic convergence. For the standard setting of nonlinear programming, where the parameter space is a Euclidean space, an algorithm's convergence typically means that every accumulation point of a sequence of iteration points, computed by the algorithm, satisfies a (usually necessary) optimality condition. Thus, every bounded sequence of iteration points would yield at least one point satisfying the optimality condition. On the other hand, if the parameter space is infinite-dimensional, then bounded sequences of iteration points might not have accumulation points, and hence a different notion of convergence is needed. Such a notion was proposed in Ref. 20, and it characterizes convergence by the condition that $\limsup_{j\to\infty} \theta(\xi_j) = 0$, where $\{\xi_j\}_{j=1}^{\infty}$ is any bounded sequence of iteration points computed by an algorithm.

The optimization problem confronting us in this paper is different. Its parameter set does not consist of a single infinite-dimensional space, but rather of the union of an infinite sequence of monotone-increasing sets. Specifically, let Φ be a given finite set; for every $N = 0, 1, 2, \ldots$, let Ξ_N be a subset of $\Phi \times \Phi \times \ldots \times \Phi$ (the set-product $N + 1$ times), and let $\Xi := \cup_{N=1}^{\infty} \Xi_N$.[7] Then Ξ is the parameter set of our optimization problem. Let Γ_N be a suitable optimality condition defined on Ξ_N, and let

[6] If Ξ is a topological space then θ is assumed to be upper-semicontinuous as well.

[7] It will become apparent from the context that there is a natural embedding of Ξ_N into Ξ_{N+1}.

θ_N be a corresponding optimality function. The algorithm presented in the next section computes a sequence ξ_j, $j = 1, 2, \ldots$, where $\xi_j \in \Xi_{N(j)}$, and $N(j+1) > N(j)$ for all $j = 1, 2, \ldots$. Its convergence will be characterized by computing either a point $\xi_j \in \Xi_{N(j)}$ such that $\theta_{N(j)}(\xi_j) = 0$ (in which case the algorithm stops), or a sequence $\{\xi_j\}_{j=1}^{\infty}$ such that $\lim_{j\to\infty} \theta_{N(j)}(\xi_j) = 0$. This characterization of convergence extends the concepts developed in Ref. 20 from the setting of an infinite-dimensional parameter space to the setting of the present paper. Its justification will be made clear once the algorithm is presented and analyzed.

We will adopt the common approach developed in Ref. 19 to proving convergence of descent algorithms. It is based on the principle of *sufficient descent*, defined as follows.

Definition 2.1. An algorithm defined on Ξ has sufficient descent with respect to the optimality functions θ_k, $k = 1, 2, \ldots$, if for every $\delta > 0$ there exists $\eta > 0$ such that, for every $j = 1, 2, \ldots$, and for every iteration point ξ_j computed by the algorithm, if $\theta_{N(j)}(\xi_j) < -\delta$, then

$$J(\xi_{j+1}) \; - \; J(\xi_j) \; < \; -\eta, \tag{1}$$

where ξ_{j+1} is the next iteration point.

Now convergence can be ensured as follows.

Proposition 2.1. Suppose that there exists a constant $D \in R$ such that $J(\xi) \geq D$ for every $\xi \in \Xi$. If a descent algorithm has the property of sufficient descent, then for every infinite sequence $\{\xi_j\}_{j=1}^{\infty}$ it computes, $\lim_{j\to\infty} \theta_{N(j)}(\xi_j) = 0$.

Proof. Suppose, for the sake of contradiction, that the algorithm computes a sequence $\{\xi_j\}_{j=1}^{\infty}$ such that $\liminf_{j\to\infty} \theta_{N(j)}(\xi_j) < 0$. Then, by Definition 2.1 there exist $\eta > 0$ and an infinite sequence $\{j_i\}_{i=1}^{\infty}$ such that, for all $i = 1, 2, \ldots$, $J(\xi_{j_i+1}) - J(\xi_{j_i}) < -\eta$. Since the algorithm is of descent, it follows that $\lim_{j\to\infty} J(\xi_j) = -\infty$, but this contradicts that fact that $J(\xi) \geq D$ for all $\xi \in \Xi$, hence completing the proof. □

3. Problem Formulation

Consider the following time-varying dynamical system,

$$\dot{x} \; = \; F(x, t), \tag{2}$$

where $x \in R^n$ is the state, $[0, T]$ is the time horizon for a given $T > 0$, the initial state $x_0 := x(0)$ is assumed given, and $F : R^n \times [0, T] \to R^n$ is a function satisfying assumptions sufficient to ensure a unique, continuous, piecewise-differentiable solution to Eq. (2). Let $L : R^n \times [0, T] \to R$ be a cost function, and consider the cost functional J, defined by

$$J \; = \; \int_0^T L(x, t)dt. \tag{3}$$

This paper considers F to be in a class of functions having the following form. Let Φ be a given, finite set of functions $f : R^n \to R^n$. Let $N \geq 0$ be any integer, and corresponding to N, let $s := (\tau_1, \ldots, \tau_N)^T \in R^N$ be any vector satisfying the inequalities

$$0 \; \leq \; \tau_1 \leq \; \ldots \; \leq \; \tau_N \; \leq \; T. \tag{4}$$

We define, for convenience, $\tau_0 := 0$ and $\tau_{N+1} := T$. Let $\{f_{\sigma(i)}\}_{i=1}^{N+1}$ be any collection of $N+1$ functions in Φ. Then, the class of functions F that we consider in Eq. (2) has the form

$$F(x, t) \; = \; f_{\sigma(i)}(x) \qquad \text{for all } t \in [\tau_{i-1}, \tau_i), \quad \text{and for all } i = 1, \ldots, N+1. \tag{5}$$

Note that, with this form of F, Eq. (2) assumes the following expression,

$$\dot{x} \; = \; f_{\sigma(i)}(x) \quad \text{for all } t \in [\tau_{i-1}, \tau_i), \quad \text{and for all } i = 1, \ldots, N+1. \tag{6}$$

To ensure a unique solution to Eq. (6) and upper bounds on its sensitivity with respect to the switching times, we make the following mild assumption.

Assumption 3.1.
(i) The functions $f \in \Phi$, and L, are continuously differentiable on R^n.
(ii) There exists a constant $K > 0$ such that, for every $x \in R^n$, and for every $f \in \Phi$,

$$||f(x)|| \ \leq \ K(||x|| + 1). \tag{7}$$

The system is called a *switched-mode hybrid system*, and the various functions $f \in \Phi$ represent its various modes, and hence they are called the *modal functions.* We define the finite sequence σ by $\sigma := \{\sigma(1), \ldots, \sigma(N+1)\}$, and refer to it as the *modal sequence.* The vector $s = (\tau_1, \ldots, \tau_N)^T$ is called the *vector of switching times*, and the pair (σ, s), denoted by ξ, is referred to as the *modal schedule,* or the *system's schedule.* Given $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\}$, it is possible that $f_{\sigma(i)} = f_{\sigma(k)}$ for $i \neq k$, in other words, a modal sequence may have a specific function $f \in \Phi$ multiple times. Consequently, we can associate a modal sequence $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\}$ with an element in the product-set Φ^{N+1}, defined as the set-product of Φ with itself $N+1$ times.

We consider the problem of minimizing J as a function of the schedule $\xi = (\sigma, s)$, where we note that N is a part of the variable σ. For future reference, we denote this problem by P. This problem may have fairly general constraints on the modal sequence σ, including, but not limited to the following type that often arises in applications: For every $f \in \Phi$, let $\Psi_f \subset \Phi$ be a set of "permissible" modal functions that are allowed to follow the function f in any modal sequence. Thus, for every $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\}$, it is required that $f_{\sigma(i+1)} \in \Psi_{f_{\sigma(i)}}$ for all $i = 1, \ldots, N$. Generally, we denote by Ψ the set of all feasible modal sequences, i.e., we require that $\sigma \in \Psi$. For a given $\sigma \in \Psi$, we also will consider the problem of minimizing J as a function of s subject to the constraints set forth in Eq. (4), and will refer to this problem by P_σ. It is a nonlinear programming problem, and we will seek a solution for it to the extent of computing a Kuhn-Tucker point. Observe that the constraint in Eq. (4) allows for the situation where $\tau_{i-1} = \tau_i$. In this case the interval $[\tau_{i-1}, \tau_i)$ is empty and hence the modal function $f_{\sigma(i)}$ plays no role in the evolution of the state trajectory via Eq. (6). However, we allow for this case to be considered since we shall bring to bear upon the problem P_σ the theory of nonlinear programming, which generally requires closed constraint sets.

Let us fix a modal sequence $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\}$ for a moment. Let $s = (\tau_1, \ldots, \tau_N)^T$ be a Kuhn-Tucker point for P_σ, and define $\xi = (\sigma, s)$. When computed by a nonlinear programming algorithm, s often is a local minimum for P_σ. However, ξ may not be a local minimum for P in the following sense. Consider a time-point $\tau \in [0, T]$ and a modal function $f \in \Phi$, henceforth denoted by $f = f_{\sigma(\star)}$. Given $\lambda > 0$, consider the insertion of the modal function $f_{\sigma(\star)}$ in the interval $[\tau - \frac{\lambda}{2}, \tau + \frac{\lambda}{2}) \cap [0, T]$, and denote the corresponding value of J as a function of $\lambda \geq 0$ by $\hat{J}(\lambda)$. Let $D_{f,\tau}(\xi)$ denote the one-sided derivative of $\hat{J}$ at $\lambda = 0$, namely, $D_{f,\tau}(\xi) = \frac{d\hat{J}}{d\lambda^+}(0)$. As shown in Ref. 16, $D_{f,\tau}(\xi)$ is well defined for all $\tau \in [0, T]$, and it is continuous except at the switching points τ_i. If $D_{f,\tau}(\xi) < 0$ then an insertion of f in a "small-enough" interval centered at τ would result in a reduction of J.

Such a mode-insertion modifies the modal sequence σ and the dimension of the switching-time vector s. For example, consider the case where $(\tau - \frac{1}{2}\lambda, \tau + \frac{1}{2}\lambda) \subset (\tau_{i-1}, \tau_i)$ for some $i = 1, \ldots, N+1$. Then the above insertion changes the schedule $\xi = (\sigma, s)$ to a new schedule, $\hat{\xi} = (\hat{\sigma}, \hat{s})$, defined by $\hat{\sigma} = \{\sigma(1), \ldots, \sigma(i), \sigma(\star), \sigma(i), \ldots, \sigma(N+1)\}$, and $\hat{s} = (\tau_1, \ldots, \tau_{i-1}, \tau - \frac{1}{2}\lambda, \tau + \frac{1}{2}\lambda, \tau_i, \ldots, \tau_N)^T$. Note that $\hat{s} \in R^{N+2}$. For an illustration, see Figure 1. Now we will be primarily interested in the case where $\lambda \to 0$, and in this case will refer to $\hat{\xi}$ as the *schedule obtained by inserting* $f = f_{\sigma(\star)}$ *to the schedule* ξ *at the time* τ.

The possibility of inserting new modes to a given schedule allows us to construct an algorithm that appends the sequencing-variable σ by first-order sensitivity information. The idea is to solve the problem P_σ for a given σ, and then to insert new modal functions g at a times t such that $D_{g,t}(\xi) < 0$. To simplify the analysis in this paper, we insert only one such a function $g \in \Phi$, and this function is determined as follows. Let us define

$$D(\xi) \ := \ \min\{D_{f,\tau}(\xi) \mid f \in \Phi \text{ such that } \hat{\sigma} \in \Psi, \ \tau \in [0, T]\}, \tag{8}$$

where $\hat{\sigma}$ is the modal sequence resulting from inserting f to ξ at time τ and (recall that) Ψ is the set

of feasible modal sequences. Let $(g,t) \in \Phi \times [0,T]$ be an argmin of the term in (8). Observe that if $D(\xi) < 0$ then the insertion of the modal function g in a small interval centered at time t would result in a reduced value of J. On the other hand, there is no indication of such a reduction in J if $D(\xi) = 0$. The case where $D(\xi) > 0$ is not possible since, for $f = f_{\sigma(i)}$ and for all $\tau \in (\tau_{i-1}, \tau_i)$, $D_{f,\tau}(\xi) = 0$, and hence, by definition (8), $D(\xi) \leq 0$ always. Based on all of this, we have the following algorithm.

Algorithm 3.1.

Data: A modal sequence σ_0.

Step 0: Set $j = 0$.

Step 1: Solve the problem P_{σ_j} to the extent of computing a switching-time vector s_j that is a Kuhn-Tucker point for P_{σ_j}. Denote the resulting schedule by $\xi_j := (\sigma_j, s_j)$.

Step 2: If $D(\xi_j) = 0$, stop and exit. Otherwise, compute $g \in \Phi$ and $t \in [0,T]$ such that $D_{g,t}(\xi_j) = D(\xi_j)$.

Step 3: Define $\hat{\xi}_{j+1} := (\sigma_{j+1}, \hat{s}_{j+1})$ to be the schedule obtained by inserting g to the schedule (σ_j, s_j) at time t.

Step 4: Set $j = j + 1$, and go to Step 1.

A few remarks are due.

Remark 3.1. We later will present a simple formula for the term $D_{f,\tau}(\xi)$, and mention straightforward extensions thereof to simultaneous insertions of multiple modal functions at a given time τ. Such multiple insertions can replace the single-mode insertion in Step 2, but we prefer to present the analysis only in the setting of the single insertion in order to keep it as simple as possible.

Remark 3.2. Even with multiple insertions, the resulting set of possible schedules does not include all possible sequences of modes. However, we use this particular modification of the modal sequences since it fits within the framework of gradient-descent.

Remark 3.3. Step 1 and Step 2 generally require infinite-loop procedures. In Step1, solving P_{σ_j} typically involves an iterative nonlinear-programming algorithm, and in Step 2, the computation of $D(\xi_j)$ involves minimization of $D_{g,t}(\xi_j)$ over the infinite set $\Phi \times [0,T]$. Implementations must involve practical stopping rules that yield approximations to the desired quantities. We do not discuss here specific approximation schemes but rather assume that they are available, and we carry out the convergence analysis by assuming exact computations.

For every $N = 0, 1, \ldots$, we define Ξ_N to be the set of schedules $\xi = (\sigma, s)$ such that $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\} \in \Psi$, and $s = (s_1, \ldots, s_N)^T$ is a Kuhn-Tucker point for P_σ. Observe that Algorithm 3.1 computes, in Step 1, a schedule $\xi_j \in \Xi_{N(j)}$ for some integer $N(j)$. Moreover, we have the inequality $N(j+1) > N(j)$ for all $j = 1, 2, \ldots$; in fact, either $N(j+1) = N(j)+2$ or $N(j+1) = N(j)+1$, depending on whether the insertion time at Step 2 was in the interior of the interval $[0,T]$ or one of its end points. We will consider these schedules to be the iteration points computed by the algorithm, and it is in their terms that we shall characterize convergence.

The condition $D(\xi_j) = 0$ can be viewed as a necessary condition for a local optimality, and hence it acts as the stopping rule in Step 2 of the algorithm. Formally, we label this condition *stationarity*, defined as follows.

Definition 3.1.

A schedule $\xi = (\sigma, s)$ is said to be a *stationary schedule* for the problem P if (i) σ is a feasible sequence, i.e., $\sigma \in \Psi$; (ii) s is a Kuhn-Tucker point for the problem P_σ; and (iii) $D(\sigma, s) = 0$.

Thus, Algorithm 3.1 stops at stationary schedules, and we expect it to converge to stationary points in the sense of Section 2. To characterize this kind of convergence, let us define the optimality function $\theta_N : \Xi_N \to R^-$ by $\theta_N(\xi) = D(\xi)$, and further defining $\Sigma_N \subset \Xi_N$ to be the set of stationary schedules, we observe that, by definition, $\theta_N(\xi) = 0$ if and only if $\xi \in \Sigma_N$. Now the main result of the paper is that, under suitable assumptions, if the algorithm computes a sequence of schedules $\xi_j \in \Xi_{N(j)}$, $j = 1, 2, \ldots$, then

$$\lim_{j \to \infty} \theta_{N(j)}(\xi_j) \;=\; 0. \tag{9}$$

Whether this result is in force depends on specific details of Algorithm 3.1, and especially on the procedure used in Step 1 to solve P_{σ_j}. This will be discussed in Section 4, where the convergence

analysis will be nontrivial due to the fact that the switching-time vectors s_j lie in successive Euclidean spaces of increasing dimensions.

Before closing this section we mention some results concerning the derivative terms $\frac{dJ}{d\tau_i}$ and $D_{f,\tau}(\xi)$, and characterizations of Kuhn-Tucker points for P_σ (see Ref. 16 for their developments). Let us fix $\sigma = \{\sigma(1), \ldots, \sigma(N+1)\}$ and consider J as a function of $s = (\tau_1, \ldots, \tau_N)^T$ satisfying Eq. (4) (s need not be a Kuhn-Tucker point for P_σ). Recall Eq. (5). For every $i = 0, \ldots, N+1$, define $x_i := x(\tau_i)$, and note that x_i is well defined since $x(\cdot)$ is continuous in t. Furthermore, define the costate vector $p(t)$ by the following equation,

$$\dot{p}(t) \;=\; -\big(\frac{\partial F}{\partial x}(x,t)\big)^T p(t) \;-\; \big(\frac{\partial L}{\partial x}(x,t)\big)^T, \tag{10}$$

with the boundary condition $p(T) = 0$, and define p_i by $p_i = p(\tau_i)$ (p_i is well defined since $p(\cdot)$ is continuous). Now the derivative $\frac{dJ}{d\tau_i}$ $(i = 1, \ldots, N)$ has the following form:[8]

$$\frac{dJ}{d\tau_i} \;=\; p_i^T\big(f_{\sigma(i)}(x_i) - f_{\sigma(i+1)}(x_i)\big). \tag{11}$$

Moreover, this derivative term is continuous in τ_i for all s satisfying the constraint inequalities in Eq. (4).

Next, consider the following characterization of Kuhn-Tucker points for P_σ. Recall that the feasible set is given by Eq. (4). Its boundary consists of points $s = (\tau_1, \ldots, \tau_N)^T$ such that $\tau_i = \tau_{i+1}$ for some $i = 0, \ldots, N$ (this includes the cases where $\tau_1 = 0$ or $\tau_N = T$ since, by definition, $\tau_0 = 0$ and $\tau_{N+1} = T$). For every $i \in \{0, \ldots, N+1\}$, define the integer-quantities $m(i)$ and $n(i)$ as follows: $m(i) := \min\{m \le i : \tau_m = \tau_i\}$, and $n(i) := \max\{n \ge i : \tau_n = \tau_i\}$. In other words, $\tau_k = \tau_i$ for all $k \in \{m(i), \ldots, n(i)\}$; if $\tau_i > 0$ then $\tau_{m(i)-1} < \tau_{m(i)}$; and if $\tau_i < T$ then $\tau_{n(i)} < \tau_{n(i)+1}$. Then, by the particular form of the inequality constraints defined by Eq. (4), it is readily seen that s is a Kuhn-Tucker point if and only if the following two equations are in force for all $i = 0, \ldots, N+1$:

$$\sum_{k=m(i)}^{i} \frac{dJ}{d\tau_k}(s) \le 0 \;\;\text{ if }\;\; \tau_i > 0, \tag{12}$$

and

$$\sum_{k=i}^{n(i)} \frac{dJ}{d\tau_k}(s) \ge 0 \;\;\text{ if }\;\; \tau_i < T. \tag{13}$$

Corollary 3.1. Let s be a Kuhn-Tucker point for P_σ. (i) If $0 < \tau_i < T$, then

$$\sum_{k=m(i)}^{n(i)} \frac{dJ}{d\tau_k}(s) \;=\; 0. \tag{14}$$

(ii) If $\tau_i = 0$, then $\sum_{k=m(i)}^{n(i)} \frac{dJ}{d\tau_k}(s) \ge 0$. (iii) If $\tau_i = T$, then $\sum_{k=m(i)}^{n(i)} \frac{dJ}{d\tau_k}(s) \le 0$.[9]

Proof. Consider the case where $0 < \tau_i < T$. Observe that $m(n(i)) = m(i)$, and hence, applying Eq. (12) to $n(i)$ in lieu of i, we obtain that $\sum_{k=m(i)}^{n(i)} \frac{dJ}{d\tau_k}(s) \le 0$. Likewise, noting that $n(m(i)) = n(i)$ and applying Eq. (13) to $m(i)$, we obtain, $\sum_{k=m(i)}^{n(i)} \frac{dJ}{d\tau_k}(s) \ge 0$. These two inequalities establish Eq. (14). The assertions (ii) and (iii) are provable in similar ways from Eqs. (12) and (13). □

Remark 3.4. We call a set of contiguous integers $\{m, \ldots, n\}$ a *block* if $m = m(i)$ and $n = n(i)$ for some $i \in \{m, \ldots, n\}$. Eq. (14) means that $\sum_{k=m}^{n} \frac{dJ}{d\tau_k}(s) = 0$ for every block $\{k, \ldots, n\} \subset \{1, \ldots, N\}$.

Finally, consider the insertion of a modal function $f \in \Phi$ at a given time $\tau \in (0, T)$. Suppose first that $\tau \in (\tau_{i-1}, \tau_i)$ for some $i = 1, \ldots, N+1$. Then (see Ref. 16), with $\xi := (\sigma, s)$, we have that

$$D_{f,\tau}(\xi) \;=\; p(\tau)^T\big(f(x(\tau)) - f_{\sigma(i)}(x(\tau))\big). \tag{15}$$

[8] We use the derivative notation $\frac{dJ}{d\tau_i}$ and not $\frac{\partial J}{\partial \tau_i}$, since we focus on the total derivative of J as a function of τ_i.

[9] Note that, in case (ii) $m(i) = 0$, and in case (iii) $n(i) = N+1$.

For the case where $\tau = \tau_i$,

$$D_{f,\tau}(\xi) \;=\; p(\tau_i)^T \Big(f(x_i) - \frac{f_{\sigma(m(i))}(x_i) + f_{\sigma(n(i)+1)}(x_i)}{2} \Big). \tag{16}$$

In the next section we establish conditions on Step 1 of Algorithm 3.1 guaranteeing convergence in the sense of Section 2.

4. **Convergence Analysis**

In order to complete the description of Algorithm 3.1 we have to specify the procedure for implementing Step 1 of that algorithm. Presented later in this section and labelled Procedure 4.1, its purpose is to solve P_{σ_j} to the extent of computing a Kuhn-Tucker point. As we shall see, it is comprised of an iterative feasible descent algorithm which starts at the point $\hat{s}_j$, computed in Step 3 of the previous iteration of Algorithm 3.1. Consequently Algorithm 3.1 is a descent algorithm as well, and hence $J(\xi_{j+1}) \leq J(\xi_j)$ for all $j = 0, 1, \ldots$. Whether it converges in the sense defined in Section 2, depends on the details of Procedure 4.1. In our initial investigation we considered a feasible-direction algorithm with Armijo stepsizes (Refs. 21 and 19), whose descent direction was against the projection of the gradient $\nabla J(s)$ onto the feasible set, defined by Eq. (4). This algorithm is known to provide sufficient descent when acting on a Euclidean space (see Refs. 21 and 19). However, in our case, although giving a descent, sufficient descent could not be proved. The stumbling block is in the increasing dimensions of the parameter spaces of the programs P_{σ_j}. The problem comes from the fact that $||\varepsilon e_N|| = \varepsilon N^{0.5}$, where e_N is the unit vector $(1, \ldots, 1)^T \in R^N$, and $\varepsilon > 0$, and this appears to deny the property of sufficient descent due to the increasing dimensions of the spaces involved.

However, there is one situation where the Armijo step size gives sufficient descent. This situation arises in the first step of Procedure 4.1. There, to guarantee sufficient descent, the procedure uses not a descent direction, but rather a *descent curve* in $R^{N(j)}$. Parameterizing it by $\lambda \geq 0$, we denote this curve by $\{C(\lambda)\}_{\lambda \geq 0}$. It is piecewise linear and continuous, and hence not differentiable everywhere. By virtue of its starting point $(\hat{s}_j)$, it will be shown to yield sufficient descent for J only in the first step of Procedure 4.1, but not necessarily in later steps, which have only descent. This, however, will be sufficient to guarantee sufficient descent of Procedure 4.1, and hence of Algorithm 3.1. Thus, the first step of Procedure 4.1 has to be specified in detail, while subsequent steps only will have to satisfy the requirements of converging to stationary points while providing descent.

To describe the first step of Procedure 4.1 and establish the relevant notation for its analysis, it is instructive to track a typical iteration of Algorithm 3.1. In Step 1 it has $\sigma_j := \{\sigma_j(1), \ldots, \sigma_j(N(j))\}$, and it solves the problem P_{σ_j} to the extent of computing a Kuhn-Tucker point $s_j := (\tau_{j,1}, \ldots, \tau_{j,N(j)})^T$. At Step 2, suppose that $D(\xi_j) < 0$, so that the algorithm computes $g \in \Phi$ and $t \in [0, T]$ such that $D_{g,t}(\xi_j) = D(\xi_j)$. Assume that $t \in (\tau_{j,i-1}, \tau_{j,i})$ for some $i = 1, \ldots, N(j) + 1$; the analysis for the case where $t = \tau_{j,i}$ is similar (as will be evident later) and hence it is omitted. Let us use the notation $g = f_{\sigma(\star)}$. Next, consider Step 3, where Figure 2 can be used as a visual aid. Here we have that

$$\sigma_{j+1} = \{\sigma_j(1), \ldots, \sigma_j(i), \sigma(\star), \sigma_j(i), \ldots, \sigma_j(N(j) + 1)\}, \tag{17}$$

and

$$\hat{s}_{j+1} \;=\; (\tau_{j,1}, \ldots, \tau_{j,i-1}, t, t, \tau_{j,i}, \ldots, \tau_{j,N(j)})^T \;\in\; R^{N(j+1)}, \tag{18}$$

where $N(j+1) = N(j) + 2$. Note that t is the time of two switching points and the modal function g is inserted between them for an interval of length 0. Moreover, if we use the notation $\hat{s}_{j+1} := (\tau_{j+1,1}, \ldots, \tau_{j+1,N(j+1)})^T$, then $\tau_{j+1,k} = \tau_{j,k}$ for all $k = 1, \ldots, i-1$; $\tau_{j+1,i} = \tau_{j+1,i+1} = t$; and $\tau_{j+1,k} = \tau_{j,k-2}$ for all $k = i+2, \ldots, N(j+1)$.

The algorithm now returns to Step 1, where it solves $P_{\sigma_{j+1}}$ by Procedure 4.1, starting from $\hat{s}_{j+1}$. The first step of this procedure consists of the Armijo step size (whose details are described below) along the curve $\{C(\lambda)\}_{\lambda \geq 0}$, next defined. For every $\lambda \geq 0$, $C(\lambda) \in R^{N(j+1)}$, and we denote its coordinates by $C(\lambda) = (c_1(\lambda), \ldots, c_{N(j+1)}(\lambda))^T$ (the dependence on $j+1$ is clear from the context and hence implicit). Its starting point is $C(0) = \hat{s}_{j+1}$ and hence $c_k(0) = \tau_{j+1,k}$ for every $k = 1, \ldots, N(j+1)$. In

particular we have that $c_k(0) = \tau_{j,k}$ for all $k = 1, \ldots, i-1$; $c_i(0) = c_{i+1}(0) = t$; and $c_k(0) = \tau_{j,k-2}$ for all $k = i+2, \ldots, N(j+1)$. The curve is defined as follows. For every $\lambda \geq 0$, $c_i(\lambda) = \max\{t - \lambda, 0\}$ and $c_{i+1}(\lambda) = \min\{t + \lambda, T\}$; for every $k = 1, \ldots, i-1$, $c_k(\lambda) = \min\{c_i(\lambda), \tau_{j+1,k}\}$; and for every $k = i+2, \ldots, N(j+1)$, $c_k(\lambda) = \max\{c_{i+1}(\lambda), \tau_{j+1,k}\}$. To put it in words, the ith and $(i+1)th$ coordinates, starting both at t, move away from each other in opposite directions at the rate of 1 until they reach 0 and T, respectively, where they stay thereafter. Along the way they "bump" into other coordinates of $\hat{s}_{j+1}$, and then they drag them along. Eventually, for λ large enough, $c_k(\lambda) = 0$ for all $k = 1, \ldots, i$, and $c_k(\lambda) = T$ for all $k = i+1, \ldots, N(j+1)$. This is a piecewise linear curve in $R^{N(j+1)}$. Initially it moves in the two-dimensional plane defined by its ith and $(i+1)st$ coordinates, and it moves in spaces of increasing dimensions each time it "bumps" into one of the coordinates of $\hat{s}_{j+1}$. We will denote the points where the curve changes directions by λ_ν, $\nu = 1, 2, \ldots$, in increasing order, and we define $\lambda_0 := 0$ for convenience.

Define the function $h : R^+ \to R$ by

$$h(\lambda) := J(C(\lambda)), \tag{19}$$

i.e., the cost functional J along the curve $C(\lambda)$. Then $h(\cdot)$ is continuous and piecewise differentiable, and it is differentiable at all but the points λ_ν, $\nu = 1, 2, \ldots$. We will denote its derivative by $h^{'}(\lambda)$ whenever it exists, and use the notation $h^{'}(\lambda^+)$ and $h^{'}(\lambda^-)$ for the right derivative and left derivative, respectively, which exist for every $\lambda \in [0, T]$.

Procedure 4.1 now has the following form.

Procedure 4.1.
Parameters: A constant $\alpha \in (0, 1)$, and a monotone-decreasing sequence of positive numbers $\{\lambda(\ell)\}_{\ell=0}^{\infty}$ such that $\lim_{\ell \to \infty} \lambda(\ell) = 0$.
Starting Point: $\hat{s}_{j+1} = C(0)$.
First Step: Compute $\bar{\ell}$ defined as

$$\bar{\ell} \ := \ \min\{\ell = 0, 1, \ldots, \ : \ h(\lambda(\ell)) - h(0) \ \leq \ \alpha\lambda(\ell)h^{'}(0^+)\}. \tag{20}$$

Define $\bar{s}_{j+1}$ by $\bar{s}_{j+1} := C(\lambda(\bar{\ell}))$.
Subsequent Steps: Use any convergent feasible descent algorithm, starting from $\bar{s}_{j+1}$, to compute s_{j+1}, a Kuhn-Tucker point for $P_{\sigma_{j+1}}$.

We recognize $\lambda(\bar{\ell})$ as the Armijo step-size along the curve $\{C(\lambda)\}$, and we refer the reader to Refs. 21 and 19 for its analysis under general assumptions a well as its practical deployment in nonlinear programming. To shed some light on the subtlety of the sufficient-descent property, we observe that generally, at any $\lambda > 0$, the direction of the curve is not against the gradient term $\nabla J(C(\lambda))$ or its projection onto the feasible set. This is the case at $\lambda = 0$, where the direction of the curve is $-\nabla J(C(0))$, but not necessarily at every $\lambda > 0$. This is a subtle point that will play a role in the forthcoming analysis: going against $-\nabla J(C(\lambda))$ for every $\lambda > 0$ may not lead to sufficient descent. Moreover, similarly constructed curves at subsequent points (other than the first point in Procedure 4.1) also may fail to provide sufficient descent. The rest of this section focuses on proving the sufficient-descent property of the first step of Procedure 4.1.

To start with the analysis, we first characterize the derivative term $h^{'}(\lambda)$. For every $\lambda \geq 0$ and for every $k = 0, \ldots, N(j+1)$, we define the integer quantities $m(\lambda; k)$ and $n(\lambda; k)$ by

$$m(\lambda; k) \ = \ \min\{m \leq k \ : \ c_m(\lambda) = c_k(\lambda)\}, \tag{21}$$

and

$$n(\lambda; k) \ = \ \max\{n \geq k \ : \ c_n(\lambda) = c_k(\lambda)\}. \tag{22}$$

Note that $m(\lambda; k)$ and $n(\lambda; k)$ are extensions of the quantities $m(i)$ and $n(i)$ defined earlier at a point s. In particular, for $\lambda = 0$, $C(0) = \hat{s}_{j+1}$, and since by assumption $\tau_{j,i-1} < t < \tau_{j,i}$, we have that $m(0; i) = m(0, i+1) = i$ and $n(0; i) = n(0, i+1) = i+1$. Observe that, for a given k, $m(\lambda; k)$ is monotone non-increasing in λ and $n(\lambda; k)$ is monotone nondecreasing in λ, and these functions change their values only at the points λ_ν, $\nu = 1, 2, \ldots$.

Next, consider the state equation Eq. (2) and the costate equation Eq. (10), which were defined for a given schedule $\xi = (\sigma, s)$. Now we have a family of schedules parameterized by λ, denoted by $\{\xi(\lambda)\}$ and defined by $\xi(\lambda) := (\sigma_{j+1}, C(\lambda))$, corresponding to which we parameterize the above equations by λ as well. Thus, for every $\lambda \geq 0$, let us denote by $F_\lambda(x,t)$ the function F in Eq. (2) corresponding to the schedule $\xi(\lambda)$, and denote by $x_\lambda(t)$ its corresponding state trajectory; namely,

$$\dot{x}_\lambda \;=\; F_\lambda(x_\lambda, t), \qquad x_\lambda(0) = x_0. \tag{23}$$

Likewise, we denote by $p_\lambda(t)$ the corresponding costate trajectory obtained by Eq. (10) with the schedule $\xi(\lambda)$, namely

$$\dot{p}_\lambda \;=\; -\big(\frac{\partial F_\lambda}{\partial x}(x_\lambda, t)\big)^T p_\lambda \;-\; \big(\frac{\partial L}{\partial x}(x_\lambda, t)\big)^T, \qquad p_\lambda(T) = 0. \tag{24}$$

Then, for a fixed $\lambda \geq 0$, in analogy with Eq. (11) we have the following equation,

$$\frac{dJ}{dc_k(\lambda)} \;=\; p_\lambda(c_k(\lambda))^T \Big(f_{\sigma_{j+1}(k)}(x_\lambda(c_k(\lambda))) - f_{\sigma_{j+1}(k+1)}(x_\lambda(c_k(\lambda))) \Big). \tag{25}$$

Let us define $\chi_{[c_k(\lambda)>0]}$ to be the characteristic (indicator) function of the event that $c_k(\lambda) > 0$, and likewise, we define $\chi_{[c_k(\lambda)<T]}$ to be the characteristic function of the event that $c_k(\lambda) < T$.

We now have the following characterization of $h^{'}(\lambda)$.

Lemma 4.1. For every $\nu = 0, 1, \ldots$, and for every $\lambda \in (\lambda_\nu, \lambda_{\nu+1})$,

$$\begin{aligned} h^{'}(\lambda) \;&=\; p_\lambda(c_i(\lambda))^T \Big(f_{\sigma_{j+1}(i+1)}(x_\lambda(c_i(\lambda))) - f_{\sigma_{j+1}(m(\lambda;i))}(x_\lambda(c_i(\lambda))) \Big) \chi_{[c_i(\lambda)>0]} \\ &+\; p_\lambda(c_{i+1}(\lambda))^T \Big(f_{\sigma_{j+1}(i+1)}(x_\lambda(c_{i+1}(\lambda))) - f_{\sigma_{j+1}(n(\lambda;i+1)+1)}(x_\lambda(c_{i+1}(\lambda))) \Big) \chi_{[c_{i+1}(\lambda)<T]}. \end{aligned} \tag{26}$$

Proof. Fix $\nu = 0, 1, \ldots$, and fix $\lambda \in (\lambda_\nu, \lambda_{\nu+1})$. For all $k = m(\lambda;i), \ldots, i$: if $c_i(\lambda) = 0$ then $c_k(\lambda) = 0$ and $\frac{dc_k(\lambda)}{d\lambda} = 0$, and if $c_i(\lambda) > 0$ then $c_k(\lambda) = t - \lambda$ and hence $\frac{dc_k(\lambda)}{d\lambda} = -1$. Similarly, for all $k = i+1, \ldots, n(\lambda; i+1)$: if $c_{i+1}(\lambda) = T$ then $c_k(\lambda) = T$ and $\frac{dc_k(\lambda)}{d\lambda} = 0$, and if $c_{i+1}(\lambda) < T$ then $c_k(\lambda) = t + \lambda$ and hence $\frac{dc_k(\lambda)}{d\lambda} = 1$. Moreover, for all $k \in \{1, \ldots, m(\lambda;i) - 1\} \cup \{n(\lambda; i+1) + 1, \ldots, N(j+1)\}$, $c_k(\lambda) = \tau_{j+1,k}$ and hence $\frac{dc_k(\lambda)}{d\lambda} = 0$. Consequently, and by an application of the chain rule, we have that

$$h^{'}(\lambda) \;=\; -\chi_{[c_i(\lambda)>0]} \sum_{k=m(\lambda;i)}^{i} \frac{dJ}{dc_k(\lambda)} \;+\; \chi_{[c_{i+1}(\lambda)<T]} \sum_{k=i+1}^{n(\lambda;i+1)} \frac{dJ}{dc_k(\lambda)}. \tag{27}$$

Plug Eq. (25) in Eq. (27) to obtain,

$$\begin{aligned} h^{'}(\lambda) \;&=\; -\chi_{[c_i(\lambda)>0]} \sum_{k=m(\lambda;i)}^{i} p_\lambda(c_k(\lambda))^T \Big(f_{\sigma_{j+1}(k)}(x_\lambda(c_k(\lambda))) - f_{\sigma_{j+1}(k+1)}(x_\lambda(c_k(\lambda))) \Big) \\ &+\; \chi_{[c_{i+1}(\lambda)<T]} \sum_{k=i+1}^{n(\lambda;i+1)} p_\lambda(c_k(\lambda))^T \Big(f_{\sigma_{j+1}(k)}(x_\lambda(c_k(\lambda))) - f_{\sigma_{j+1}(k+1)}(x_\lambda(c_k(\lambda))) \Big). \end{aligned} \tag{28}$$

By definition of $m(\lambda;i)$ and $n(\lambda;i+1)$, we have that $c_k(\lambda) = c_i(\lambda)$ for all $k = m(\lambda;i), \ldots, i$, and $c_k(\lambda) = c_{i+1}(\lambda)$ for all $k = i+1, \ldots, n(\lambda;i+1)$. Using this in Eq. (28) renders the two sums telescopic and hence yields Eq. (26). □

Recall that $m(\lambda;k) = m(\lambda_\nu;k)$ and $n(\lambda;k) = n(\lambda_\nu;k)$ for all $\lambda \in (\lambda_\nu, \lambda_{\nu+1})$. Therefore $h^{'}(\lambda)$ is discontinuous only at the points λ_ν as can be seen from the presence of the terms $m(\lambda;i)$ and $n(\lambda;i+1)$ in Eq. (26). Consequently $h^{'}(\lambda_\nu^+)$ is given by Eq. (26) with $\lambda = \lambda_\nu$, namely,

$$\begin{aligned} h^{'}(\lambda_\nu^+) \;&=\; p_{\lambda_\nu}(c_i(\lambda_\nu))^T \Big(f_{\sigma_{j+1}(i+1)}(x_{\lambda_\nu}(c_i(\lambda_\nu))) - f_{\sigma_{j+1}(m(\lambda_\nu;i))}(x_{\lambda_\nu}(c_i(\lambda_\nu))) \Big) \chi_{[c_i(\lambda_\nu)>0]} \\ &+\; p_{\lambda_\nu}(c_{i+1}(\lambda_\nu))^T \Big(f_{\sigma_{j+1}(i+1)}(x_{\lambda_\nu}(c_{i+1}(\lambda_\nu))) - f_{\sigma_{j+1}(n(\lambda_\nu;i+1)+1)}(x_{\lambda_\nu}(c_{i+1}(\lambda_\nu))) \Big) \chi_{[c_{i+1}(\lambda_\nu)<T]}. \end{aligned} \tag{29}$$

Of a particular interest is the case $\nu = 0$, namely the term $h^{'}(0^+)$, for which we have the following formula.

Lemma 4.2. The derivative term $h^{'}(0^+)$ has the following form,

$$h^{'}(0^+) \;=\; 2D(\xi_j). \tag{30}$$

Proof. Recall that $C(0) = \hat{s}_{j+1}$, and hence, and by Eq. (18), $c_i(0) = t$ and $c_{i+1}(0) = t$. Also, since by assumption (made at the start of this discussion) $t \in (\tau_{j,i-1}, \tau_{j,i})$, we have that $m(0; i) = i$ and $n(0; i + 1) = i + 1$. Therefore, we get from Eq. (29) that

$$h^{'}(0^+) \;= \\ p_0(t)^T\Big(f_{\sigma_{j+1}(i+1)}(x_0(t)) - f_{\sigma_{j+1}(i)}(x_0(t))\Big) \;+\; p_0(t)^T\Big(f_{\sigma_{j+1}(i+1)}(x_0(t)) - f_{\sigma_{j+1}(i+2)}(x_0(t))\Big). \tag{31}$$

By Eq. (17), $f_{\sigma_{j+1}(i)} = f_{\sigma_j(i)}$, $f_{\sigma_{j+1}(i+1)} = g$, and $f_{\sigma_{j+1}(i+2)} = f_{\sigma_j(i)}$. Plug this in Eq. (31) to obtain,

$$h^{'}(0^+) \;=\; 2p_0(t)^T\Big(g(x_0(t)) - f_{\sigma_j(i)}(x_0(t))\Big). \tag{32}$$

Finally, Eq. (30) follows from an application of Eq. (15) with g and t instead of f and τ, and the fact that $D_{g,t}(\xi_j) = D(\xi_j)$. □

We next turn to proving convergence of Algorithm 3.1. The proof is based on the following successive steps.

Step 1. We first establish that the functions $x_\lambda(\cdot)$ and $p_\lambda(\cdot)$, viewed as elements in $L^\infty[0, T]$, are Lipschitz continuous in λ.

Step 2. We prove that $h^{'}(\lambda^+)$ has a continuity property, uniformly with respect to the schedule ξ_j, at $\lambda = 0$. This is despite the fact that $h^{'}(\lambda)$ is generally discontinuous in λ. What makes this statement true is the way the point $C(0) = \hat{s}_{j+1}$ was constructed in Step 3 of Algorithm 3.1.

Step 3. We prove that Algorithm 3.1 with Procedure 4.1 in its first step has the sufficient descent property.

The proofs progress through a sequence of preliminary results.

Lemma 4.3. There exists a constant $D > 0$ such that, for every schedule $\hat{\xi}_{j+1}$ computed in Step 3 of Algorithm 3.1, and for every $\lambda \geq 0$, $||x_\lambda||_\infty \leq D$ and $||p_\lambda||_\infty \leq D$, where $||\cdot||_\infty$ denotes the L^∞ norm of a function.

Proof. Immediate from Eqs. (6) and (10), Assumption 3.1, and the fact that the set Φ of modal functions is finite. □

Lemma 4.4. There exists a constant $L_1 > 0$ such that, for every schedule $\hat{\xi}_{j+1}$ computed in Step 3 of Algorithm 3.1, for every $\lambda > 0$, and for every $\tau \in [0, T)$ and $\delta > 0$, $||x_\lambda(\tau + \delta) - x_\lambda(\tau)|| \leq L_1\delta$ and $||p_\lambda(\tau + \delta) - p_\lambda(\tau)|| \leq L_1\delta$.

Proof. Follows directly from Eqs. (6) and (10) (or Eqs. (23) and (24), their λ-dependent variants), Assumption 3.1, and Lemma 4.3. □

Lemma 4.5. There exists a constant $L > 0$ such that, for every schedule $\hat{\xi}_{j+1}$ computed by Algorithm 3.1, and for every $\lambda > 0$, $||x_\lambda - x_0||_\infty \leq L\lambda$ and $||p_\lambda - p_0||_\infty \leq L\lambda$.

The proof follows directly from Lemma 4.3 in conjunction with standard arguments in sensitivity analysis of solutions to ordinary differential equations; see Ref. 19, Section 5.6.

The key technical argument is developed in the following lemma.

Lemma 4.6. There exists a constant $K > 0$ such that, for every schedule $\hat{\xi}_{j+1}$ computed by Algorithm 3.1, and for every $\lambda > 0$,

$$|h^{'}(\lambda^+) - h^{'}(0^+)| \;\leq\; K\lambda. \tag{33}$$

Proof. Fix $\lambda > 0$. We can assume, without loss of generality, that $c_i(\lambda) > 0$ and $c_{i+1}(\lambda) < T$. Then, applications of Eq. (26) and Eq. (29) with $\nu = 0$ yield,

$$h^{'}(\lambda^+) - h^{'}(0^+)$$

$$
\begin{aligned}
&= p_\lambda(c_i(\lambda))^T \Big(f_{\sigma_{j+1}(i+1)}\big(x_\lambda(c_i(\lambda))\big) \;-\; f_{\sigma_{j+1}(m(\lambda;i))}\big(x_\lambda(c_i(\lambda))\big) \Big) \\
&+ p_\lambda(c_{i+1}(\lambda))^T \Big(f_{\sigma_{j+1}(i+1)}\big(x_\lambda(c_{i+1}(\lambda))\big) \;-\; f_{\sigma_{j+1}(n(\lambda;i+1)+1)}\big(x_\lambda(c_{i+1}(\lambda))\big) \Big) \\
&- p_0(c_i(0))^T \Big(f_{\sigma_{j+1}(i+1)}\big(x_0(c_i(0))\big) \;-\; f_{\sigma_{j+1}(m(0;i))}\big(x_0(c_i(0))\big) \Big) \\
&- p_0(c_{i+1}(0))^T \Big(f_{\sigma_{j+1}(i+1)}\big(x_0(c_{i+1}(0))\big) \;-\; f_{\sigma_{j+1}(n(0;i+1)+1)}\big(x_0(c_{i+1}(0))\big) \Big). \qquad (34)
\end{aligned}
$$

Defining U_1, U_2, U_3, and U_4 by Eqs. (36)-(39), below, and rearranging the various terms in Eq. (34), we obtain that

$$h^{'}(\lambda^+) - h^{'}(0^+) \;=\; U_1 - U_2 + U_3 - U_4, \qquad (35)$$

where

$$U_1 \;=\; p_\lambda(c_i(\lambda))^T f_{\sigma_{j+1}(i+1)}\big(x_\lambda(c_i(\lambda))\big) \;-\; p_0(c_i(0))^T f_{\sigma_{j+1}(i+1)}\big(x_0(c_i(0))\big), \qquad (36)$$

$$U_2 \;=\; p_\lambda(c_i(\lambda))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_\lambda(c_i(\lambda))\big) \;-\; p_0(c_i(0))^T f_{\sigma_{j+1}(m(0;i))}\big(x_0(c_i(0))\big), \qquad (37)$$

$$U_3 \;=\; p_\lambda(c_{i+1}(\lambda))^T f_{\sigma_{j+1}(i+1)}\big(x_\lambda(c_{i+1}(\lambda))\big) \;-\; p_0(c_{i+1}(0))^T f_{\sigma_{j+1}(i+1)}\big(x_0(c_{i+1}(0))\big), \qquad (38)$$

$$U_4 \;=\; p_\lambda(c_{i+1}(\lambda))^T f_{\sigma_{j+1}(n(\lambda;i+1)+1)}\big(x_\lambda(c_{i+1}(\lambda))\big) \;-\; p_0(c_{i+1}(0))^T f_{\sigma_{j+1}(n(0;i+1)+1)}\big(x_0(c_{i+1}(0))\big). \qquad (39)$$

Recall that $c_i(0) = t$ and $c_i(\lambda) = t - \lambda$, and hence $c_i(\lambda) - c_i(0) = -\lambda$. Therefore, and by Lemma 4.5, Lemma 4.4, and Lemma 4.3, there exists a constant $K_1 > 0$ (independent of $\hat{\xi}_{j+1}$ or λ) such that,

$$|U_1| \;\leq\; K_1\lambda. \qquad (40)$$

Similarly, $c_{i+1}(0) = t$ and $c_{i+1}(\lambda) = t + \lambda$, and hence $c_{i+1}(\lambda) - c_{i+1}(0) = \lambda$; therefore there exists a constant $K_3 > 0$ such that,

$$|U_3| \;\leq\; K_3\lambda. \qquad (41)$$

We need similar inequalities for U_2 and U_4. These, however, are more problematic. The reason, regarding U_2, can be seen in the RHS of Eq. (37), whose two additive terms contain different modal functions, namely $f_{\sigma_{j+1}(m(\lambda;i))}$ and $f_{\sigma_{j+1}(m(0;i))}$, and hence their difference does not indicate a bound like the one in Eq. (40). A similar issue arises in Eq. (39). However, we shall see that such bounds are indeed in force, and this is due to the particular structure of the algorithm, and especially to the fact that the point s_j was a Kuhn-Tucker point for P_{σ_j}.

Let us consider U_2 as defined in Eq. (37). Adding and subtracting the same terms, and defining $U_{2,1}$, $U_{2,2}$, and $U_{2,3}$ by Eqs. (43)-(45), below, we have that

$$U_2 \;=\; U_{2,1} \;+\; U_{2,2} \;+\; U_{2,3}. \qquad (42)$$

The various terms in the RHS of Eq. (42) are defined as follows.

$$U_{2.1} \;=\; p_\lambda(c_i(\lambda))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_\lambda(c_i(\lambda))\big) \;-\; p_0(c_i(0))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_0(c_i(0))\big), \qquad (43)$$

$$U_{2,2} \;=\; p_0(c_i(0))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_0(c_i(0))\big) \;-\; p_0(c_{m(\lambda;i)}(0))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_0(c_{m(\lambda;i)}(0))\big), \qquad (44)$$

$$U_{2,3} \;=\; p_0(c_{m(\lambda;i)}(0))^T f_{\sigma_{j+1}(m(\lambda;i))}\big(x_0(c_{m(\lambda;i)}(0))\big) \;-\; p_0(c_i(0))^T f_{\sigma_{j+1}(m(0;i))}\big(x_0(c_i(0))\big). \qquad (45)$$

Regarding $U_{2.1}$, similarly to Eqs. (40) and (41), there exists a constant $K_{2,1} > 0$ such that

$$|U_{2.1}| \;\leq\; K_{2,1}\lambda. \qquad (46)$$

Concerning $U_{2,2}$, recall that $c_i(0) = t$; $c_i(\lambda) = t - \lambda$; by Eq. (21) $c_{m(\lambda;i)}(\lambda) = c_i(\lambda)$; $m(\lambda;i) \leq i$ and hence $c_{m(\lambda;i)}(0) \leq c_i(0)$; and $c_{m(\lambda;i)}(\cdot)$ is monotone non-increasing in λ. Therefore, we have that $0 \leq c_i(0) - c_{m(\lambda;i)}(0) \leq c_i(0) - c_{m(\lambda;i)}(\lambda) = t - (t - \lambda) = \lambda$, and hence, $|c_i(0) - c_{m(\lambda;i)}(0)| \leq \lambda$. Consequently and by Eq. (44), in a way similar to Eqs. (40) and (41), there exists a constant $K_{2,2} > 0$ such that

$$|U_{2,2}| \;\leq\; K_{2,2}\lambda. \qquad (47)$$

$U_{2,3}$ is where the problem lies, because the two additive terms in the RHS of Eq. (45) involve two different modal functions. Recall that $m(0;i)=i$, and hence (and by Eq. (45)) we can write $U_{2,3}$ as

$$U_{2,3} \;=\; \sum_{k=m(\lambda;i)}^{i-1} \Big(p_0(c_k(0))^T f_{\sigma_{j+1}(k)}\big(x_0(c_k(0))\big) \;-\; p_0(c_{k+1}(0))^T f_{\sigma_{j+1}(k+1)}\big(x_0(c_{k+1}(0))\big)\Big). \tag{48}$$

Note that, in the special case where $m(\lambda;i)=i$, the range of the sum in Eq. (48) is vacuous and hence Eq. (48) suggests that $U_{2,3}=0$, which is consistent with Eq. (45). Thus, we can assume without loss of generality that $m(\lambda;i)<i$. Defining $V_{2,3,k}$ and $W_{2,3,k}$ by

$$V_{2,3,k} \;=\; p_0(c_k(0))^T \Big(f_{\sigma_{j+1}(k)}\big(x_0(c_k(0))\big) \;-\; f_{\sigma_{j+1}(k+1)}\big(x_0(c_k(0))\big)\Big) \tag{49}$$

and

$$W_{2,3,k} \;=\; p_0(c_k(0))^T f_{\sigma_{j+1}(k+1)}\big(x_0(c_k(0))\big) \;-\; p_0(c_{k+1}(0))^T f_{\sigma_{j+1}(k+1)}\big(x_0(c_{k+1}(0))\big), \tag{50}$$

we have that

$$U_{2,3} \;=\; \sum_{k=m(\lambda;i)}^{i-1} \Big(V_{2,3,k} + W_{2,3,k}\Big). \tag{51}$$

Recall Eq. (18) that for every $k=m(\lambda;i),\ldots,i-1$, $c_k(0)=\tau_{j,k}$, where $s_j=(\tau_{j,1},\ldots,\tau_{j,N(j)})^T$ was the Kuhn-Tucker point for P_{σ_j} last computed in Step 1 of Algorithm 3.1. Moreover, by applying Eq. (25) with $\lambda=0$ we obtain, for all $k=m(\lambda;i),\ldots,i-1$, that

$$V_{2,3,k} \;=\; \frac{dJ}{d\tau_{j,k}}(s_j). \tag{52}$$

Recall the definition of a *block*, made in Remark 3.4. We now show that for every $\lambda>0$, the set $\{m(\lambda;i),\ldots,i-1\}$ is comprised of the union of contiguous blocks associated with $C(0)$. To this end, all we need to show is that $c_{m(\lambda;i)-1}(0)<c_{m(\lambda;i)}(0)$ and $c_{i-1}(0)<c_i(0)$, since in this case $m(\lambda;i)-1$ is not in the block containing $m(\lambda;i)$, and $i-1$ is not in the block containing i, according to the curve $C(0)$. The latter inequality follows from our assumption that $\tau_{j,i-1}<t<\tau_{j,i}$. Regarding the former inequality, by definition Eq. (21), $c_{m(\lambda;i)-1}(\lambda)<c_{m(\lambda;i)}(\lambda)$; and by the definition of the curve, $c_{m(\lambda;i)-1}(\lambda)=c_{m(\lambda;i)-1}(0)$. This implies that $c_{m(\lambda;i)-1}(0)<c_{m(\lambda;i)}(\lambda)$. Since $c_{m(\lambda;i)}(\lambda)\le c_{m(\lambda;i)}(0)$, it follows that $c_{m(\lambda;i)-1}(0)<c_{m(\lambda;i)}(0)$.

Now Corollary 3.1 and the remark following it, together with with Eq. (52) and the fact that the set $\{m(\lambda;i),\ldots,i-1\}$ is the union of blocks associated with $C(0)$, imply that

$$\sum_{k=m(\lambda;i)}^{i-1} V_{2,3,k} \;=\; 0. \tag{53}$$

Finally, consider the term $W_{2,3,k}$, defined in Eq. (50). Similarly to the proof of of Eq. (40), there exists a constant $K_{2,3}>0$ such that for every $k=m(\lambda;i),\ldots,i-1$,

$$|W_{2,3,k}| \;\le\; K_{2,3}\big(c_{k+1}(0)-c_k(0)\big). \tag{54}$$

But $c_i(0)=t$ and $c_{m(\lambda;i)}(0)\ge c_{m(\lambda;i)}(\lambda)\ge t-\lambda$, and hence $c_i(0)-c_{m(\lambda;i)}(0)\le\lambda$. Consequently, (and since $c_i(0)-c_{m(\lambda;i)}(0)\ge 0$) we have that

$$\sum_{k=m(\lambda;i)}^{i-1} |W_{2,3,k}| \;\le\; K_{2,3} \sum_{k=m(\lambda;i)}^{i-1} \big(c_{k+1}(0)-c_k(0)\big) \;=\; K_{2,3}\big(c_i(0)-c_{m(\lambda;i)}(0)\big) \;\le\; K_{2,3}\lambda. \tag{55}$$

Putting it all together we obtain, by Eqs. (51), (53), and (55), that

$$|U_{2,3}| \;\le\; K_{2,3}\lambda. \tag{56}$$

Defining $K_2 := K_{2,1} + K_{2,2} + K_{2,3}$, Eqs. (42), (46), (47), and Eq. (56) imply that

$$|U_2| \;\leq\; K_2\lambda. \tag{57}$$

In a similar way (see Eq. (39)), there exists a constant $K_4 > 0$ such that,

$$|U_4| \;\leq\; K_4\lambda. \tag{58}$$

Al last, by Eqs. (35), (40), (57), (41), and (58), we have that $|h^{'}(\lambda^+) - h^{'}(0^+)| \leq (K_1 + K_2 + K_3 + K_4)\lambda$. Defining $K := K_1 + K_2 + k_3 + k_4$, Eq. (33) follows. □

Remark 4.1. The main argument of the proof is based on the bounds derived in Eqs. (40), (41), (46), (47), and Eq. (56). All but the latter one are applications of the Lipschitz continuity of $x_\lambda(t)$ and $p_\lambda(t)$ in λ and t (see Lemma 4.4 and Lemma 4.5). On the other hand, Eq. (56) is based on Eq. (53), and this equation is due to the particular starting point of the curve $\{C(\lambda)\}$, namely $\hat{s}_{j+1}$. Starting the curve from any other point generally would not yield an equation like Eq. (53) since the derivative term $h^{'}(\lambda)$ is discontinuous. Consequently, a descent algorithm may not yield sufficient descent, and Algorithm 3.1 might not converge in the sense of Section 2.

There is another subtlety that is ironed out by the particular choice of the curve $\{C(\lambda)\}$. The crucial equation Eq. (53) is true regardless of the number $i - m(\lambda; i)$. A different descent curve, even though starting from $\hat{s}_{j+1}$, might have a positive (but not zero) upper bound on the term $\sum_{k=m(\lambda;i)}^{i-1} V_{2,3,k}$ in the LHS of Eq. (53). This bound may be "small" but grow with $i - m(\lambda; i)$. The latter number, in turn, may grow with the dimension of the curve, $N(j+1)$. This, in turn, could prevent the sufficient-descent property and hence the convergence of Algorithm 3.1. For this reason we were unable to prove convergence of a feasible gradient-descent algorithm in lieu of Procedure 4.1, nor do we believe that it is true. Descent we certainly could get but without its "sufficient" qualification, and this is due to the growing dimensions of the successive problems P_{σ_j}. The particular choice of the curve $\{C(\lambda)\}$ not only provides a measure of continuity of the derivative $h^{'}(\lambda)$ at the starting point of the curve, but also avoids the problem associated with its growing dimensionality in successive iterations.

The next lemma pertains to the first step of Procedure 4.1. Recall that Algorithm 3.1 enters Procedure 4.1 with a point $\hat{\xi}_{j+1}$, and Procedure 4.1 then computes the point $\bar{\xi}_{j+1}$ in its first step.

Lemma 4.7. For every $\delta > 0$ there exists $\eta > 0$ such that, if Algorithm 3.1 enters Procedure 4.1 with $\hat{\xi}_{j+1}$ such that $h^{'}(0^+) < -\delta$, then, for the computed point $\bar{\xi}_{j+1}$,

$$J(\bar{\xi}_{j+1}) - J(\hat{\xi}_{j+1}) \;\leq\; -\eta. \tag{59}$$

Proof. The proof is based on standard arguments in the analysis of gradient-descent algorithms. Let Procedure 4.1 start at a point $\hat{\xi}_{j+1}$. Fix $\delta > 0$ and suppose that $h^{'}(0^+) < -\delta$. Recall $\alpha \in (0, 1)$ as given by Procedure 4.1, and let K be the Lipschitz constant given by Lemma 4.6 (Eq. (33)). By the mean value theorem, for every $\lambda \geq 0$ there exists $\zeta \in conv\{h^{'}(\tilde{\lambda}^+) | \tilde{\lambda} \in [0, \lambda]\}$ ($conv(\cdot)$ indicates convex hull) such that $h(\lambda) - h(0) = \zeta\lambda$. Since by Eq. (33) we have that $|\zeta - h^{'}(0^+)| \leq K\lambda$, we have that

$$h(\lambda) - h(0) \;\leq\; \big(h^{'}(0^+) + K\lambda\big)\lambda. \tag{60}$$

Define $\bar{\lambda} := (1 - \alpha)\delta/K$. Since $h^{'}(0^+) < -\delta$ by assumption, it follows that for every $\lambda \in [0, \bar{\lambda}]$,

$$h^{'}(0^+) + K\lambda \;\leq\; h^{'}(0^+) + K\bar{\lambda} \;=\; h^{'}(0^+) + (1 - \alpha)\delta \;<\; h^{'}(0^+) - (1 - \alpha)h^{'}(0^+) \;\leq\; \alpha h^{'}(0^+).$$

Therefore, and by Eq. (60), we have that $h(\lambda) - h(0) \leq \alpha\lambda h^{'}(0^+)$. An examination of Eq. (20) reveals that $\bar{\ell}$ is bounded from below by $\hat{\ell} := \min\{\ell = 0, 1, \ldots \;:\; \lambda(\ell) < \bar{\lambda}\}$, and by Eq. (20) and the fact that the sequence $\{\lambda(\ell)\}$ is monotone decreasing, $h(\lambda(\hat{\ell})) - h(0) \leq \alpha\lambda(\hat{\ell})h^{'}(0^+) \leq -\alpha\lambda(\hat{\ell})\delta$. Defining $\eta := \alpha\lambda(\hat{\ell})\delta$ and recalling the definition of $\bar{s}_{j+1}$ (following Eq. (20)) and the fact that $h(\lambda) := J(C(\lambda))$, Eq. (59) follows. □

Finally, we present the main result of the paper.

Theorem 4.1. Suppose that Algorithm 3.1, with Procedure 4.1 for its Step 1, computes a sequence of schedules $\{\xi_j\}_{j=1}^{\infty}$. Then,

$$\lim_{j\to\infty} D(\xi_j) \;=\; 0. \tag{61}$$

Proof. By Eq. (3) and Lemma 4.3, J is bounded from below. By Procedure 4.1, Algorithm 3.1 is a descent algorithm and hence $J(\xi_{j+1}) \leq J(\xi_j)$ for all $j = 1, 2, \ldots$. Fix $\delta > 0$. By Lemma 4.2 and Lemma 4.7 there exists $\eta > 0$ such that, for every $j = 1, 2, \ldots$, if $D(\xi_j) < -\delta$ then $J(\xi_{j+1}) - J(\xi_j) \leq -\eta$. Consequently Algorithm 3.1 has the sufficient descent property with the optimality function $\theta(\xi_j) := D(\xi_j)$, and by Proposition 2.1, Eq. (61) is satisfied. □

We point out that this theorem guarantees only stationarity of the sequence of schedules, computed by the algorithm, in the sense of Definition 3.1. If a limiting schedule exists we can expect it to be a local minimum because the algorithm has the descent property. However, to compute a global minimum one may consider several runs of the algorithm from various different initial points (schedules).

5. Illustrative Example

Consider the problem of controlling the fluid level in a tank by adjusting the input flow rate to an auxiliary tank, as shown in Figure 3. This particular system was addressed in Ref. 22, where a state-feedback timing control law was designed to switch between two modes (a PID controller and a minimum-time controller) in order to track a target reference fluid level in the second tank. The problem considered here is different, primarily because we compute an open-loop optimal control. The control-input variable consists of the fluid inflow rate into the upper tank, denoted by $u(t)$. The state variables, $x_1(t)$ and $x_2(t)$, consist of the fluid levels at the upper tank and the lower tank, respectively, and we define $x(t) \in R^2$ by $x(t) = (x_1(t), x_2(t))^T$. By Torricelli's principle, the state equation has the following form, for given (fixed) values of $\gamma_1 > 0$ and $\gamma_2 > 0$.

$$\dot{x} = f(x, u) = \begin{bmatrix} -\gamma_1\sqrt{(x_1)} + u \\ \gamma_1\sqrt{(x_1)} - \gamma_2\sqrt{(x_2)} \end{bmatrix}. \tag{62}$$

Furthermore, suppose that the input valve can be in either one of three states: fully open, half open, or closed. Correspondingly, the input flow rate $u(t)$ attains one of the following three values, u_{max}, $\frac{1}{2}u_{max}$, or 0, for some given value of $u_{max} > 0$.

Corresponding to the three values of $u(t)$, we denote the right-hand side of Eq. (62) by $f_1(x)$ when $u = u_{max}$, $f_2(x)$ when $u = \frac{1}{2}u_{max}$, and $f_3(x)$ when $u = 0$. Consequently, $\Phi = \{f_1, f_2, f_3\}$, and we assume that there are no restrictions or constraints on the sequencing of the modal functions.

Given an initial state $x_0 = x(0)$ and a final time $T > 0$, the objective of our switching-control strategy is to have the fluid level in the lower tank track a given reference signal, denoted by $x_r(t)$. Thus, the performance functional that we minimize is

$$J = K \int_0^T (x_2(t) - x_r(t))^2 dt, \tag{63}$$

for a suitable constant $K > 0$.

We applied Algorithm 3.1 to minimize the cost functional J, with the parameters $x_0 = (0.4, 0.4)^T$, $T = 5$, $K = 10$, $\gamma_1 = \gamma_2 = 1$, $u_{max} = 1$, and the reference signal $x_r(t) = 0.5 + 0.25\frac{t}{T}$. In the first run of the algorithm we chose $\sigma_0 = \{1\}$, namely, the input valve is fully open throughout the time horizon $[0, 5]$. Due to practical considerations, we stopped the algorithm-run when the condition $D(\xi_j) > -0.01$ was satisfied, instead of the condition $D(\xi_j) \geq 0$ that is specified in Step 2 of the algorithm. All the differential equations were solved by a forward Euler method with step size of $\Delta t = 0.001$, and the minimum in the computation of $D(\xi_j)$ (Step 2, Eq. (8)) was computed on a uniform grid on the time interval $[0, 5]$ with increments of 0.001. In Procedure 4.1, we chose $\alpha = 0.5$ and $\lambda(\ell) = (0.5)^\ell$. For all steps in the main loop other than the first, we used the Armijo step size in the direction against the projection of the cost's gradient onto the feasible set. This procedure was made to stop whenever the magnitude of the descent direction was less than 0.01.

The results of the algorithm's run are shown in Figure 4 and Figure 5. Figure 4(a) shows the cost value and the dimension of the switching-time vector as functions of the number of gradient descent steps. Figure 4(b) plots $D(\xi_j)$ as a function of the iteration count, j, of Algorithm 3.1. Algorithm 3.1 stops at the $7th$ iteration, where the cost value is 0.105. Figure 5 shows the values of the input

function and the state trajectories at the final iteration. Figure 5(a) plots the schedule of the modal functions as a function of time, and Figure 5(b) plots $x_1(t)$ (dashed), $x_2(t)$ (solid), and the reference signal $x_r(t)$ (charred). It can be seen from Figure 5(b) that the fluid level in the lower tank indeed tracks the reference signal.

Figure 5(a) indicates that, according to the final schedule, the input valve switches between the states of closed and fully open, but it is never half open. To verify that this solution is indeed locally optimal, we tested it on the same optimal control problem except that the control is continuous and it can assume any value in the interval $[0, 1]$, namely $u(t) \in [0, 1]$ for every $t \in [0, T]$. The hamiltonian for this problem has the following form (with the costate $p := (p_1, p_2)^T$),

$$H(x, u, p) \;=\; p_1 u + (p_2 - p_1)\sqrt{x_1} - p_2\sqrt{x_2} + 10(x_2 - x_r)^2, \tag{64}$$

which indicates that the optimal control is bang-bang. We verified that the control shown in Figure 5(a) satisfies Pontryagin's maximum principle for 96% of the time-points $t \in [0, T]$, which indicates that it is indeed close to being locally optimal.

6. Conclusions.

This paper presents an algorithm for solving optimal scheduling problems on switched-mode hybrid dynamical systems. The scheduling variable consists of a discrete parameter and a continuous parameter. The discrete parameter is comprised of the sequence of the modes deployed during the system's operation, and the continuous parameter consists of the switching times among the modes. The continuous parameter can be handled by standard nonlinear-programming optimization techniques, but the discrete parameter is more problematic. To overcome this difficulty, the problem is cast as an optimal-control problem, upon which variational techniques are brought to bear. In particular, we handle the discrete parameter by using first-order approximation principles to insert a new mode to an existing schedule. The algorithm alternates between two phases: in the inner phase it optimizes the cost functional with respect to the continuous parameter for a given sequence of modes, and in the outer phase it adds new modes to the resulting schedule.

The variational principles underlying the mode insertion allow us to use non-linear programming algorithms for solving the optimal scheduling problem. However, the optimization problem is not defined over a single linear space, but rather over the union of an infinite-dimensional Euclidean spaces of increasing dimensions. This is a somewhat unusual setting for optimization, and therefore the paper first defines appropriate notions of stationarity and convergence. Subsequently, much of the analysis focuses on proving convergence of the proposed algorithm by using the concept of sufficient descent.

References

[1] LINCOLN, B., and RANTZER, A., *Optimizing Linear Systems Switching*, Proceedings of the 40th IEEE Conference on Decision and Control, Orlando, Florida, pp. 2063–2068, 2001.

[2] REHBINDER, H., and SANFIRDSON, M., *Scheduling of a Limited Communication Channel for Optimal Control*, Proceedings of the 39th IEEE Conference on Decision and Control, Sidney, Australia, pp. 1011-1016, 2000.

[3] WALSH, G., YE, H., and BUSHNELL, L., *Stability Analysis of Networked Control Systems*, Proceedings of the American Control Conference, San Diego, California, pp. 2876–2880, 1999.

[4] BROCKETT, R., *Stabilization of Motor Networks*, Proceedings of the 34th IEEE Conference on Decision and Control, New Orleans, Louisiana, pp. 1484–1488, 1995.

[5] EGERSTEDT, M., and WARDI, Y., *Multiprocess Control Using Queuing Theory*, Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, pp. 1991-1996, 2002.

[6] HRISTU-VARSAKELIS, D., *Feedback Control Systems as Users of Shared Network: Communication Sequences that Guarantee Stability*, Proceedings of the 40th IEEE Conference on Decision and Control, Orlando, Florida, pp. 3631–3631, 2001.

[7] SHAIKH, M.S., and CAINES, P., *On Trajectory Optimization for Hybrid Systems: Theory and Algorithms for Fixed Schedules*, Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, pp. 1997-1998, 2002.

[8] SHAIKH, M.S., and CAINES, P.E., *On the Optimal Control of Hybrid Systems: Optimization of Trajectories, Switching Times and Location Schedules*, Proceedings of the 6th International Workshop on Hybrid Systems: Computation and Control, Prague, Czech Republic, pp. 466-481, 2003.

[9] XU, X., and ANTSAKLIS, P., *Optimal Control of Switched Autonomous Systems*, Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, pp. 4401-4406, 2002.

[10] XU, X., and ANTSAKLIS, P.J., *Optimal Control of Switched Systems via Nonlinear Optimization Based on Direct Differentiations of Value Functions*, International Journal of Control, Vol. 75, pp. 1406-1426, 2002.

[11] PICCOLI, B. *Hybrid Systems and Optimal Control*, Proceedings of the 37th IEEE Conference on Decision and Control, Tampa, Florida, pp. 13-18, 1998.

[12] SUSSMANN, H.J., *A Maximum Principle for Hybrid Optimal Control Problems*, Proceedings of the 38th IEEE Conference on Decision and Control, Phoenix, Arizona, pp. 425-430, 1999.

[13] SUSSMANN, H.J., *Set-Valued Differentials and the Hybrid Maximum Principle*, Proceedings of the 39th IEEE Conference on Decision and Control, Sydney, Australia, pp. 558 -563, 2000.

[14] ALAMIR, M., and ATTIA, S.A., *On Solving Optimal Control Problems for Switched Nonlinear Systems by Strong Variations Algorithms*, Proceedings of the 6th IFAC Symposium on Nonlinear Control Systems, Stuttgart, Germany, pp. 558 -563, 2004.

[15] ATTIA, S.A., ALAMIR, M., and CANUDAS DE Wit, C., *Sub Optimal Control of Switched Nonlinear Systems Under Location and Switching Constraints*, Proceedings of the 16th IFAC World Congress, Prague, Czech Republic, 2005.

[16] EGERSTEDT, M., WARDI, Y., and AXELSSON, H., *Transition-Time Optimization for Switched Systems* IEEE Transactions on Automatic Control, Vol. -51, pp. 110-115, 2006.

[17] AXELSSON, H., WARDI, Y., and EGERSTEDT, M., *Transition-Time Optimization for Switched Systems*, Proceedings of the 16th IFAC World Congress, Prague, Czech Republic, , 2005.

[18] AXELSSON, H., WARDI, Y., EGERSTEDT, M., and VERRIESTt, E.I., *A Provably Convergent Algorithm for Transition-Time Optimization in Switched Systems* Proceedings of the 44th IEEE Conference on Decision and Control, Sevilla, Spain, pp. 558 -563, 2005.

[19] POLAK, E., *Optimization Algorithms and Consistent Approximations*, Springer-Verlag, New York, NY, 1997.

[20] POLAK, E., and WARDI, Y., *A Study of Minimizing Sequences*, SIAM Journal on Control and Optimization, Vol. 22, pp. 599-609, 1984.

[21] ARMIJO, L., *Minimization of Functions Having Lipschitz Continuous First-Partial Derivatives*, Pacific Journal of Mathematics, Vol. 16, pp. 1-3, 1966.

[22] MALMBORG, J., and EKER, J., *Hybrid Control of a Double Tank System*, Proceedings of the 1997 Conference on Control Applications, Hartford, Connecticut, pp. 133-138, 1997.