Optimal Control of Switching Times in Switched Linear Systems

Bartolomeo Stellato, Sina Ober-Blöbaum and Paul J. Goulart

Abstract— **Switching time optimization arises in finite-horizon optimal control for switched systems where, given a sequence of continuous dynamics, we minimize a cost function with respect to the switching times. In this paper we propose an efficient method for computing optimal switching times in switched linear systems. We derive simple expressions for the cost function, the gradient and the Hessian which can be computed efficiently online without performing any integration. With the proposed method, the most expensive computations are decomposed into independent scalar exponentials which can be efficiently computed and parallelized. Simulation results show that our method is able to provide fast convergence and handle efficiently a high number switching times.**

I. Introduction

Hybrid models commonly arise in several engineering problems where systems with continuous dynamics interact with discrete events. Optimal control of hybrid systems presents several challenges because it involves both continuous and discrete decisions which make the corresponding optimization problems NP-hard to solve [1]. Switched systems are a particular class of hybrid systems consisting of several continuous subsystems where a switching law defines the active one at each time instant; see [2] for a recent survey.

In the present work we focus on optimal control of autonomous switched systems. In particular, given a sequence of state dynamics, we study the problem of computing the optimal switching instants at which the dynamics have to change in order to minimize a predefined cost function. This problem is usually referred to as *switching time optimization*.

This topic has been studied extensively in the last decade. In [3] the authors provide a method to construct an offline mapping of the optimal switching times from the initial state of the system. The switching order is given by a so called "master algorithm" solving a integer optimization problem at a higher level. Even if the switching time optimization is reduced to a simple function evaluation, this approach suffers from the high storage requirements typical of explicit control approaches [4] as the dimension of the system and the number of switchings increase.

More recent approaches focus on finding optimal switching times using iterative optimization methods. In [5] an expression for the gradient of the cost function with respect to the switching times is derived for the case of nonlinear systems. A first-order method based on Armijo step sizes is then adopted to find the optimal switching times. An extension for discrete-time nonlinear systems is given in [6]. However, first order methods are very sensitive to the problem data and they can exhibit slow convergence [7]. In [8] an expression for the Hessian of the cost function is derived for nonlinear dynamics and a second-order method is adopted to find the optimal switching times showing significant improvements on the number of iterations compared to the first-order method in [5]. However, both these first and second-order approaches suffer from the computational complexity of multiple numerical integrations required to solve the differential equations used to define the cost function, the gradient and the Hessian (in the second-order case). Note that the Hessian definition in [8] requires an additional number of integrations to be performed.

To tackle the computational complexity, in [9] the switching time optimization problem for linear time-varying dynamics is formulated so that only a set of differential equations needs to be solved before the optimization procedure. Once the integration is performed, the steepest descent direction can be computed directly without solving any differential equation. However, in [9] no closed form expression for the Hessian of the cost function is provided and only a steepest descent algorithm is adopted.

In this work we present an efficient method to formulate and solve switching time optimization problems for switched linear systems. We develop efficiently computable expressions for the cost function, the gradient and the Hessian with respect to the switching times. We show that the most expensive computations required to evaluate the cost function and its first and second derivatives are the same and can be greatly simplified through precomputations. With the proposed method, there is no need to perform any integration and the most expensive operations are just independent scalar exponentials which can be efficiently computed and parallelized.

We also proposed an extension of this work to nonlinear systems in [10] where the optimization variables are the switching intervals. The results in [10] have been implemented in the Julia package `SwitchTimeOpt.jl` leading to up to two orders of magnitude improvements with respect to the results appearing in this paper.

The rest of the paper is organized as follows. Section II defines the switching time optimization problem. In Section III the state evolution and its first and second derivatives are presented. In Section IV we derive the expressions for the cost function, the gradient and the Hessian. Efficient methods to perform the operations shared between $J(\tau)$, $\nabla J(\tau)$ and $H_J(\tau)$ are provided in Section V. Extensions of our method are shown in Section VI. We provide two numerical examples in Section VII and conclusions in Section VIII.

II. Problem Statement

Consider a switched autonomous linear system switching N times in the time window $t \in [0, T]$. Let us define a switching sequence $\{\tau_i\}_{i=1}^N$ described by the vector $\tau = \begin{bmatrix}\tau_1 \ \dots \ \tau_N\end{bmatrix}^\top \in \mathbb{R}^N$. To simplify the derivations we

This work was supported by the People Programme (Marie Curie Actions) of the European Unions Seventh Framework Programme (FP7/2007-2013) under REA grant agreement no 607957 (TEMPO). B. Stellato, S. Ober-Blöbaum and P. J. Goulart are with the University of Oxford, Parks Road, Oxford, OX1 3PJ, U.K. {`bartolomeo.stellato`, `sina.ober-blobaum`, `paul.goulart`}`@eng.ox.ac.uk`.

define the extrema of our time interval as $\tau_0 := 0$ and $\tau_{N+1} := T$. Our goal is to find, given a sequence of switched dynamics, the optimal switching times $\tau^\star$ minimizing the cost function

$$J(\tau) = \frac{1}{2}\int_0^T \|x(t,\tau)\|_Q^2 \mathrm{d}t = \frac{1}{2}\int_0^T x(t,\tau)^\top Q x(t,\tau)\mathrm{d}t, \tag{1}$$

where $x(t,\tau) \in \mathbb{R}^{n_x}$ is the state of the system. Note that τ_0 and τ_{N+1} are not optimization variables. $Q = Q^\top \in \mathrm{S}_+^{n_x}$ is a symmetric positive semidefinite matrix.

The optimal switching times problem can be written as

$$\begin{aligned} &\text{minimize} && J(\tau) \\ &\text{subject to} && \dot{x}(t) = A_i x(t),\ t \in [\tau_i, \tau_{i+1}),\ i = 0,\dots,N \\ &&& x(0,\tau) = x_0 \\ &&& \tau \in \mathcal{T}(0,T). \end{aligned} \tag{2}$$

The initial state of the system is $x_0 \in \mathbb{R}^{n_x}$. The set $\mathcal{T}(0,T)$ is the truncated monotone cone of dimension N

$$\mathcal{T}(0,T) = \left\{\tau \in \mathbb{R}^N \mid 0 \leq \tau_1 \leq \cdots \leq T\right\}. \tag{3}$$

Note that consecutive switching times are allowed to coincide when the optimal solution neglects certain dynamics and modifies the switching order. This idea has been proposed in [11] and we will adapt it to our method in Section VI.

The function $J(\tau)$ is in general not convex, but it is continuously differentiable [12] and its first and second derivatives can be used efficiently within a SQP or Interior Point method to obtain locally optimal switching times. In order to obtain a real-time implementable algorithm, we will derive tractable formulations of the cost function $J(\tau)$, its gradient $\nabla J(\tau)$ and its Hessian $H_J(\tau)$ that exploit the particular linear structure.

III. State Evolution and Derivatives

The state trajectory at time t can be computed starting from switching instant τ_i with $T \geq t \geq \tau_i$ as

$$x(t,\tau) = \Phi(t,\tau_i,\tau)x(\tau_i,\tau), \tag{4}$$

where Φ is the state transition matrix defined for this switched system as

$$\Phi(t,\tau_i,\tau) := e^{A_\ell(t-\tau_\ell)}e^{A_{\ell-1}(\tau_\ell-\tau_{\ell-1})}\cdots e^{A_i(\tau_{i+1}-\tau_i)}, \tag{5}$$

where ℓ is the last switching before time t. The initial state is defined as $x(0,\tau) := x_0$.

Lemma 1 (First Derivative of State Evolution)**.** *The derivative of the state with respect to switching time τ_i with $i = 1,\dots,N$ can be written in terms of the state at time τ_i as*

$$\frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i} = \begin{cases} 0 & t < \tau_i \\ \Phi(t,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau) & t \geq \tau_i. \end{cases} \tag{6}$$

Proof. For $t < \tau_i$ the state trajectory does not depend on the switching time, hence the derivative is zero. For $t \geq \tau_i$, it is necessary to differentiate Equation (4) using the chain rule

$$\begin{aligned} \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i} &= \frac{\mathrm{d}\Phi(t,\tau_i,\tau)}{\mathrm{d}\tau_i}x(\tau_i,\tau) + \Phi(t,\tau_i,\tau)\frac{\mathrm{d}x(\tau_i,\tau)}{\mathrm{d}\tau_i} \\ &= -A_i\Phi(t,\tau_i,\tau)x(\tau_i,\tau) + \Phi(t,\tau_i,\tau)A_{i-1}x(\tau_i,\tau) \\ &= \Phi(t,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau), \end{aligned}$$

where in the second equality we made use of the definition of the state transition matrix from (5). ■

In order to implement a second-order optimization technique such as SQP or Interior Point methods, we also require the second derivative.

Lemma 2 (Second Derivative of State Evolution)**.** *The second derivative of the state evolution $x(t,\tau)$ with respect to switching instants τ_i and τ_j with $i,j = 1,\dots,N$ is*

$$\begin{aligned} &\frac{\mathrm{d}^2x(t,\tau)}{\mathrm{d}\tau_i\mathrm{d}\tau_j} = \\ &= \begin{cases} 0 & t < \tau_j \\ \Phi(t,\tau_i,\tau)(A_{i-1}^2 + A_i^2 + \\ \qquad - 2A_iA_{i-1})x(\tau_i,\tau) & \textit{if } t \geq \tau_j \wedge i = j \\ \Phi(t,\tau_j,\tau)(A_{j-1}-A_j)\Phi(\tau_j,\tau_i,\tau)\cdot \\ \qquad \cdot(A_{i-1}-A_i)x(\tau_i,\tau) & \textit{if } t \geq \tau_j \wedge i < j. \end{cases} \end{aligned} \tag{7}$$

Proof. To obtain the second derivative we differentiate Equation (6) with respect to τ_j. If $t < \tau_j$, the state trajectory, and hence its derivative, do not depend on τ_j. Thus, in that case the second derivative is zero. When $t \geq \tau_j$, we will consider separately the two cases when i and j coincide or not. When $i = j$, we apply the chain rule to (6) obtaining

$$\begin{aligned} \frac{\mathrm{d}^2x(t,\tau)}{\mathrm{d}\tau_i^2} &= \frac{\mathrm{d}\Phi(t,\tau_i,\tau)}{\mathrm{d}\tau_i}(A_{i-1}-A_i)x(\tau_i,\tau) + \\ &\qquad + \Phi(t,\tau_i,\tau)(A_{i-1}-A_i)\frac{\mathrm{d}x(\tau_i,\tau)}{\mathrm{d}\tau_i} \\ &= \Phi(t,\tau_i,\tau)(A_{i-1}^2 + A_i^2 - 2A_iA_{i-1})x(\tau_i,\tau), \end{aligned}$$

where the second equality comes from the definition of the state transition matrix in (5). In the case when $i \neq j$, we consider only $i < j$ since the case when $i > j$ can be derived by swapping i and j. We can differentiate Equation (6) with respect to τ_j to obtain

$$\begin{aligned} \frac{\mathrm{d}^2x(t,\tau)}{\mathrm{d}\tau_j\mathrm{d}\tau_i} &= \frac{\mathrm{d}}{\mathrm{d}\tau_j}\left(\Phi(t,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau)\right) \\ &= \frac{\mathrm{d}}{\mathrm{d}\tau_j}\left(\Phi(t,\tau_j,\tau)\Phi(\tau_j,\tau_i,\tau)\right)(A_{i-1}-A_i)x(\tau_i,\tau) \\ &= \Phi(t,\tau_j,\tau)(A_{j-1}-A_j)\Phi(\tau_j,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau). \end{aligned}$$

■

IV. Cost Function, Gradient and Hessian

Having obtained the state evolution derivatives with respect to the switching times in Section III, we are now in the position to compute the required derivatives of the objective function $J(\tau)$.

Before computing the function and its derivative we need to define the following matrix expression, which we will use heavily in the remainder of the paper.

Definition 1. *Given a switching sequence τ and two instants τ_a, τ_b with $\tau_a \leq \tau_b$, we define $P(\tau_b,\tau_a,\tau) \in \mathbb{S}_+^n$ being the following symmetric positive semidefinite matrix*

$$P(\tau_b,\tau_a,\tau) := \int_{\tau_a}^{\tau_b} \Phi(t,\tau_a,\tau)^\top Q\Phi(t,\tau_a,\tau)\mathrm{d}t, \tag{8}$$

where $\Phi(t,\tau_a,\tau)$ is the state transition matrix defined in (5).

The cost function $J(\tau)$ is a quadratic function of the initial state x_0 and, by plugging (4) in (1), can be written as

$$J(\tau) = \frac{1}{2}x_0^\top P(T,0,\tau)x_0 \tag{9}$$

where we made use of Definition 1.

A. Gradient and Hessian Derivations

In the following two results we will derive the expressions for the gradient and the Hessian of the cost function.

Proposition 1 (Gradient $\nabla J(\tau)$). *The gradient of the cost function $J(\tau)$*

$$\nabla J(\tau) = \begin{bmatrix} \frac{\mathrm{d}J(\tau)}{\mathrm{d}\tau_1} & \dots & \frac{\mathrm{d}J(\tau)}{\mathrm{d}\tau_N} \end{bmatrix}^\top \tag{10}$$

can be computed as

$$\begin{aligned} \nabla J(\tau)_i &= \frac{\mathrm{d}J(\tau)}{\mathrm{d}\tau_i} \\ &= x(\tau_i,\tau)^\top P(T,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau). \end{aligned} \tag{11}$$

Proof. Let us focus on rewriting the first derivative of the cost function with respect to a single switching time τ_i.

$$\begin{aligned} \frac{\mathrm{d}J(\tau)}{\mathrm{d}\tau_i} &= \frac{\mathrm{d}}{\mathrm{d}\tau_i}\left(\frac{1}{2}\int_0^{\tau_i} \|x(t,\tau)\|_Q^2 \mathrm{d}t\right) + \\ &\qquad \frac{\mathrm{d}}{\mathrm{d}\tau_i}\left(\frac{1}{2}\int_{\tau_i}^{T} \|x(t,\tau)\|_Q^2 \mathrm{d}t\right) \\ &= \frac{1}{2}\|x(\tau_i,\tau)\|_Q^2 \\ &\quad - \frac{1}{2}\|x(\tau_i,\tau)\|_Q^2 + \int_{\tau_i}^{T} \frac{\mathrm{d}}{\mathrm{d}\tau_i}\left(\frac{1}{2}\|x(t,\tau)\|_Q^2\right)\mathrm{d}t \\ &= \int_{\tau_i}^{T} x(t,\tau)^\top Q \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i}\mathrm{d}t. \end{aligned} \tag{12}$$

The first term has been rewritten using the fundamental theorem of calculus by noting that $x(t,\tau)$ does not depend on τ_i for $t < \tau_i$, while the second one using the Leibniz rule for integral differentiation. We can now plug in the state evolution (4) and its first derivative with respect to τ_i from (6) obtaining

$$\begin{aligned} \frac{\mathrm{d}J(\tau)}{\mathrm{d}\tau_i} &= x(\tau_i,\tau)^\top \Big(\int_{\tau_i}^{T} \Phi(t,\tau_i,\tau)^\top Q \cdot \\ &\qquad \cdot \Phi(t,\tau_i,\tau)\mathrm{d}t\Big)(A_{i-1}-A_i)x(\tau_i,\tau) \\ &= x(\tau_i,\tau)^\top P(T,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau), \end{aligned}$$

which completes the proof. ■

Remark 1. *Note that $p(\tau_i) = P(T,\tau_i,\tau)x(\tau_i,\tau)$ corresponds to the costate definition in [5]. This relationship can be seen by noting that $p(T) = 0$ and taking its derivative with respect to τ_i making use of the Leibniz rule for integral differentiation.*

$$\frac{\mathrm{d}p(\tau_i)}{\mathrm{d}\tau_i} = -A_i^\top p(\tau_i) - Qx(\tau_i,\tau), \tag{13}$$

which corresponds to Equation (11) in [5].

Proposition 2 (Hessian $H_J(\tau)$). *The Hessian $H_J(\tau)$ of the cost function can be computed as*

$$\begin{aligned} H_J(\tau)_{i,j} &= \frac{\mathrm{d}^2 J(\tau)}{\mathrm{d}\tau_i \mathrm{d}\tau_j} \\ &= \begin{cases} x(\tau_i,\tau)^\top \Big(-Q(A_{i-1}-A_i) + \\ \quad + (A_{i-1}-A_i)^\top P(T,\tau_i,\tau)(A_{i-1}-A_i) + \\ \quad\quad + P(T,\tau_i,\tau) \cdot \\ \quad\quad\quad \cdot (A_{i-1}^2 + A_i^2 - 2A_iA_{i-1})\Big)x(\tau_i,\tau), & \text{if } i = j \\ x(\tau_i,\tau)^\top (A_{i-1}-A_i)^\top \Phi(\tau_j,\tau_i,\tau)^\top \cdot \\ \quad \cdot \Big(P(T,\tau_j,\tau)(A_{j-1}-A_j) + \\ \quad + (A_{j-1}-A_j)^\top P(T,\tau_j,\tau)\Big)x(\tau_j,\tau) & \text{if } i < j \end{cases} \end{aligned} \tag{14}$$

where $i,j = 1,\dots,N$.

Proof. Let us consider first the case in which $i = j$. The second derivative of the cost function can be computed by differentiating (12) with respect to τ_i

$$\begin{aligned} \frac{\mathrm{d}^2 J(\tau)}{\mathrm{d}\tau_i^2} &= \frac{\mathrm{d}}{\mathrm{d}\tau_i}\left(\int_{\tau_i}^{T} x(t,\tau)^\top Q \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i}\mathrm{d}t\right) \\ &= -x(\tau_i,\tau)^\top Q \frac{\mathrm{d}x(\tau_i,\tau)}{\mathrm{d}\tau_i} + \\ &\quad + \int_{\tau_i}^{T} \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i}^\top Q \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i} + \\ &\qquad + x(t,\tau)^\top Q \frac{\mathrm{d}^2 x(t,\tau)}{\mathrm{d}\tau_i^2}\mathrm{d}t, \end{aligned}$$

where we applied the Leibniz rule for integral differentiation and the chain rule. We can now combine the first (6) and the second (7) derivatives of the state trajectory obtaining

$$\begin{aligned} \frac{\mathrm{d}^2 J(\tau)}{\mathrm{d}\tau_i^2} &= -x(\tau_i,\tau)^\top Q(A_{i-1}-A_i)x(\tau_i,\tau) + \\ &\quad + x(\tau_i,\tau)^\top (A_{i-1}-A_i)^\top \cdot \\ &\cdot \left(\int_{\tau_i}^{T} \Phi(t,\tau_i,\tau)^\top Q \Phi(t,\tau_i,\tau)\mathrm{d}t\right)(A_{i-1}-A_i)x(\tau_i,\tau) + \\ &\quad + x(\tau_i,\tau)^\top \left(\int_{\tau_i}^{T} \Phi(t,\tau_i,\tau)^\top Q \Phi(t,\tau_i,\tau)\mathrm{d}t\right) \cdot \\ &\qquad \cdot (A_{i-1}^2 + A_i^2 - 2A_iA_{i-1})x(\tau_i,\tau) \\ &= x(\tau_i,\tau)^\top \Big(-Q(A_{i-1}-A_i) + \\ &\quad + (A_{i-1}-A_i)^\top P(T,\tau_i,\tau)(A_{i-1}-A_i) + \\ &\quad + P(T,\tau_i,\tau)(A_{i-1}^2 + A_i^2 - 2A_iA_{i-1})\Big)x(\tau_i,\tau). \end{aligned}$$

In the last equality we made use of Definition 1.

When $i < j$, we consider the first derivative (12) with respect to τ_j and we differentiate it with respect to τ_i

$$\begin{aligned} \frac{\mathrm{d}^2 J(\tau)}{\mathrm{d}\tau_i \mathrm{d}\tau_j} &= \frac{\mathrm{d}}{\mathrm{d}\tau_i}\left(\int_{\tau_j}^{T} x(t,\tau)^\top Q \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_j}\mathrm{d}t\right) \\ &= \int_{\tau_j}^{T} \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_i}^\top Q \frac{\mathrm{d}x(t,\tau)}{\mathrm{d}\tau_j} + x(t,\tau)^\top Q \frac{\mathrm{d}^2 x(t,\tau)}{\mathrm{d}\tau_i \mathrm{d}\tau_j}\mathrm{d}t. \end{aligned}$$

We can now apply the first (6) and the second (7) derivatives of the state trajectory obtaining

$$\begin{aligned}\frac{\mathrm{d}^2 J(\tau)}{\mathrm{d}\tau_i \mathrm{d}\tau_j} &= x(\tau_i,\tau)^\top (A_{i-1}-A_i)^\top \Phi(\tau_j,\tau_i,\tau)^\top \cdot \\ &\quad \cdot \left(\int_{\tau_j}^{T} \Phi(t,\tau_j,\tau)^\top Q \Phi(t,\tau_j,\tau)\mathrm{d}t\right)\cdot \\ &\quad \cdot (A_{j-1}-A_j)x(\tau_j,\tau) + \\ &\quad + x(\tau_j,\tau)^\top \left(\int_{\tau_j}^{T} \Phi(t,\tau_j,\tau)^\top Q \Phi(t,\tau_j,\tau)\mathrm{d}t\right)\cdot \\ &\quad \cdot (A_{j-1}-A_j)\Phi(\tau_j,\tau_i,\tau)(A_{i-1}-A_i)x(\tau_i,\tau) \\ &= x(\tau_i,\tau)^\top (A_{i-1}-A_i)^\top \Phi(\tau_j,\tau_i,\tau)^\top \big(P(T,\tau_j,\tau)\cdot \\ &\quad \cdot (A_{j-1}-A_j) + (A_{j-1}-A_j)^\top P(T,\tau_j,\tau)\big)x(\tau_j,\tau).\end{aligned}$$

Where the first equality comes from decomposing $\Phi(t,\tau_i,\tau) = \Phi(t,\tau_j,\tau)\Phi(\tau_j,\tau_i,\tau)$. The second equality above is obtained by transposing the second term of the sum (which is a scalar) and by using Definition 1. ■

V. Operations Shared by $J(\tau)$, $\nabla J(\tau)$ and $H_J(\tau)$

Regardless of the second-order optimization method employed, many of the numerical operations needed to compute $J(\tau), \nabla J(\tau)$ and $H_J(\tau)$ at each iteration are the same. Thus, it is necessary to perform these computations only once per iteration.

A. Matrix Exponentials

For several of the operations described in Section IV we need to compute the matrix exponentials for each interval of the switching sequence

$$\mathcal{E}_i = e^{A_i(\tau_{i+1}-\tau_i)}, \quad i = 0,\ldots,N. \tag{15}$$

Even though computing a matrix exponential is not a trivial operation to perform at every iteration step, the matrices $A_i,\ i = 0,\ldots,N$ are always the same and only the time intervals change.

We assume that matrices A_i are diagonalizable (i.e. semi-simple) in the form $A_i = T_i^\top \Lambda_i T_i$, where $T_i \in \mathbb{R}^{n_x \times n_x}$ are the nonsingular matrices of right eigenvectors and $\Lambda_i \in \mathbb{R}^{n_x \times n_x}$ are the diagonal matrices of the eigenvalues. Note that if the matrices are not semi-simple it is possible to apply small perturbations in order to obtain nonsingular T_i matrices. We can then, precompute the matrices decomposition T_i and T_i^{-1} and obtain the matrix exponentials at each iteration as simple scalar exponentials of the elements on the diagonal of Λ_i:

$$\mathcal{E}_i = T_i^{-1} e^{\Lambda_i(\tau_{i+1}-\tau_i)} T_i. \tag{16}$$

Note that the scalar exponentials are independent and can be computed in parallel to minimize the computation times.

B. Matrices P

To obtain the cost function and its first and second derivatives, we need to compute matrix $P(T,\tau_i,\tau)$ for $i = 0,\ldots,N$ as defined in (1).

We can decompose the integral in a sum for every interval between consecutive switching times

$$\begin{aligned} P(T,\tau_i,\tau) &= \sum_{k=i}^{N} \int_{\tau_k}^{\tau_{k+1}} \Phi(t,\tau_i,\tau)^\top Q \Phi(t,\tau_i,\tau)\mathrm{d}t \\ &= \sum_{k=i}^{N} \Phi(\tau_k,\tau_i,\tau)^\top \bigg(\int_{\tau_k}^{\tau_{k+1}} \Phi(t,\tau_k,\tau)^\top Q \cdot \\ &\qquad \cdot \Phi(t,\tau_k,\tau)\mathrm{d}t \bigg) \Phi(\tau_k,\tau_i,\tau). \end{aligned} \tag{17}$$

We now define the matrices solutions of the matrix exponentials between parenthesis as M_k

$$\begin{aligned} M_k &:= \int_{\tau_k}^{\tau_{k+1}} \Phi(t,\tau_k,\tau)^\top Q \Phi(t,\tau_k,\tau)\mathrm{d}t \\ &= \int_0^{\tau_{k+1}-\tau_k} e^{A_k^\top \eta} Q e^{A_k \eta}\mathrm{d}\eta, \end{aligned}$$

where we applied the substitution $\eta = t - \tau_k$. We can rewrite the integral by parts as

$$\begin{aligned} e^{A_k^\top \eta} Q e^{A_k \eta}\Big|_0^{\tau_{k+1}-\tau_k} &= A_k^\top \left(\int_0^{\tau_{k+1}-\tau_k} e^{A_k^\top \eta} Q e^{A_k \eta}\mathrm{d}\eta \right) + \\ &\quad + \left(\int_0^{\tau_{k+1}-\tau_k} e^{A_k^\top \eta} Q e^{A_k \eta}\mathrm{d}\eta \right) A_k. \end{aligned} \tag{18}$$

It is straightforward to notice that matrices M_k can be computed by solving the Lyapunov equation

$$A_k^\top M_k + M_k A_k + \tilde{Q}_k = 0 \tag{19}$$

with $\tilde{Q}_k = Q - \mathcal{E}_k^\top Q \mathcal{E}_k$. Equation (19) can be solved as a linear system by vectorizing the matrices using the $\operatorname{vec}(\cdot)$ operator. By noting the property of the vector operator of the product of three matrices $\operatorname{vec}(ABC) = (C^\top \otimes A)\operatorname{vec}(B)$ where $\otimes$ denotes the Kronecker product, we can rewrite (19) as

$$(I_{n_x} \otimes A_k^\top + A_k^\top \otimes I_{n_x})\operatorname{vec}(M_k) = -\operatorname{vec}(\tilde{Q}_k), \tag{20}$$

where I_{n_x} is the identity matrix of dimension n_x. We can now combine (19) and solve (20) obtaining

$$\operatorname{vec}(M_k) = \mathcal{K}_k^{-1}\operatorname{vec}\left(\mathcal{E}_k^\top Q \mathcal{E}_k - Q\right), \tag{21}$$

where $\mathcal{K}_k^{-1} = (I_{n_x} \otimes A_k^\top + A_k^\top \otimes I_{n_x})^{-1}$.

Observe that the solution to (20) exists and is unique if and only if the matrices $A_k^\top$ and $-A_k^\top$ have no common eigenvalues [13]. Thus, we can invert matrix $\mathcal{K}_k$ if and only if A_k has no zero eigenvalues, i.e. is nonsingular. In practice we can add small perturbations to matrices A_k to make sure the inversion is stable.

Matrices $\mathcal{K}_k^{-1}$ depend only on individual linear dynamics k and can be precomputed offline. After computing the matrix exponentials $\mathcal{E}_k$, online we need only to compute the vector $\operatorname{vec}\left(\mathcal{E}_k^\top Q \mathcal{E}_k - Q\right)$ and multiply it by $\mathcal{K}_k^{-1}$ as in Equation (21) to obtain M_k for $k = 0,\ldots,N$.

Finally, we can compute matrices $P(T,\tau_i,\tau)$. From (5), matrix $\Phi(\tau_k,\tau_i,\tau)$ in (17) is just a product of the matrix exponentials precomputed in (16): $\Phi(\tau_k,\tau_i,\tau) = \prod_{p=i}^{k-1}\mathcal{E}_p$ if $k > i$ or $\Phi(\tau_i,\tau_i,\tau) = I_{n_x}$. Note that the matrices in the product are multiplied right to left with increasing index p.

By noting that $P(T,\tau_N,\tau) = M_N$, we can compute all the other matrices matrices recursively with $i = N-1,\ldots,0$:

$$P(T,\tau_i,\tau) = M_i + \mathcal{E}_i^\top P(T,\tau_{i+1},\tau)\mathcal{E}_i. \tag{22}$$

C. Operations Required at Every Iteration

Exploiting the precomputation in Sections V-A and V-B we require the following operations to be performed at each iteration of the optimization algorithm:

1) Compute matrix exponentials $\mathcal{E}_i$ for $i = 0, \dots, N$ as in Equation (16). This operation requires only scalar exponentials and two matrix multiplications for every i.
2) Compute the Lyapunov equations' solutions M_k for $k = 0, \dots, N$ as in Equation (21). This operation involves just three matrix products and a subtraction for every k.
3) Compute integrals $P(T, \tau_i, \tau)$ for $i = 0, \dots, N$ using Equation (22) consisting in two matrix multiplications and one addition for every i.

The cost function $J(\tau)$, the gradient $\nabla J(\tau)$ and the Hessian $H_J(\tau)$ can be then computed using Equations (9), (11) and (14) by simple matrix multiplications and additions.

Note that at each iteration the value of the state at the switching times can be directly computed using the recursion: $x(\tau_{i+1}, \tau) = \mathcal{E}_i x(\tau_i, \tau)$ with $i = 0, \dots, N$.

VI. Extensions

A. Affine Models

Affine models of the form

$$\dot{x}(t) = A_i x(t) + f_i, \quad t \in [\tau_i, \tau_{i+1}), \quad i = 0, \dots, N, \quad (23)$$

can be casted as switched linear models by augmenting the state-space from $\mathbb{R}^{n_x}$ to $\mathbb{R}^{n_x+1}$, see [3]. The augmentation can be written as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{\tilde{x}}(t) \end{bmatrix} = \begin{bmatrix} A_i & f_i \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \tilde{x}(t) \end{bmatrix}, \quad t \in [\tau_i, \tau_{i+1}), \quad i = 0, \dots, N,$$

with $\tilde{x}(0) = 1$. The weighting matrix Q has to be replaced by matrix $\tilde{Q} = \text{blkdiag}(Q, 0)$. This augmentation causes the dynamics matrices to become singular, which bring problems in the inversion in Equation (21). However, as noted in Section V, we can perturb the newly obtained dynamics to make sure the computations are numerically stable. Note that dynamical models in this form encompass a wide class of systems including linear systems with integer inputs.

B. Changing Switching Order

The feasible set defined in (3) is a truncated monotone cone where the switching instants keep a predefined switching sequence. For this feasible set, consecutive switching times are allowed to coincide, i.e. $\tau_{i+1} = \tau_i$, meaning that the dynamics directly switch from $i-1$ to $i+1$ without passing through i. This behavior allows the solver to discard some dynamics and adapt the switching sequence without recourse to integer optimization problems.

Following the idea in [11], this observation allows us to obtain the optimal switching sequence online by extending the optimization problem. Given $N+1$ possible continuous dynamics $i = 0, \dots, N$ switching without any predefined order, we can construct a bigger problem with $n_s(N+1) - 1$ switching times composed of n_s consecutive switching sequences from dynamics $i = 0$ to N; see Figure 1.

In this way the solver has access to all the possible switching combinations of the dynamics to find a local optimum without any constraint on the switching order. Note that this approach is not guaranteed to find the optimal switching sequence since the problem is not convex and we are computing a locally optimal solution.

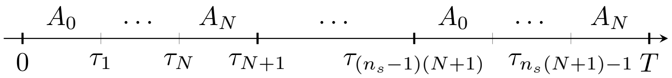

Fig. 1: Repeating switching times sequence allowing to change the ordering of the dynamics

However, this extension gives a much higher degree of flexibility compared to methods with a single predefined switching sequence iterating over the dynamics. Moreover, thanks to our inexpensive formulations of the cost function and its derivatives, even though the number of switching times increases, this approach presents lower computational burden compared to methods where mixed-integer control problems need to be solved to find the optimal sequence.

VII. Examples

To demonstrate the implementation ease and the efficiency of our method we program our definitions of the cost function and its first and second derivatives as function callbacks to the Interior Point solver IPOPT [14] on MATLAB on a Macbook Pro with Intel Core i7. No integration routine is required. Note that the implementation could be made even more efficient by writing the function callbacks in C++.

A. Unstable Switched Dynamics

Consider the switched system from [9] described by the two unstable dynamics

$$A_1 = \begin{bmatrix} -1 & 0 \\ 1 & 2 \end{bmatrix} \quad \text{and} \quad A_2 = \begin{bmatrix} 1 & 1 \\ 1 & -2 \end{bmatrix}. \quad (24)$$

Note that A_1 and A_2 have no common eigenvectors. The system transitions happen $N = 5$ times between 0 and $T = 1$ according to the modes sequence $\{1, 2, 1, 2, 1, 2\}$ and the cost function matrix is $Q = I_2$. Without any integration routine our approach converges to precision 10^{-8} in 8 iterations to the optimal switching times $\tau^\star = \begin{bmatrix} 0.100 & 0.297 & 0.433 & 0.642 & 0.767 \end{bmatrix}^\top$ which correspond to the same solution obtained in [9]. The total execution time is $175\,\text{ms}$ of which $75\,\text{ms}$ used by the IPOPT routine and $100\,\text{ms}$ by MATLAB functions callbacks.

B. One Phase of a Three-Level Inverter with RL-load

Consider a simplification of the power inverter model in [15] where only a single-phase is considered attached to the neutral point N. We assume the DC-link voltage V_{dc} to be constant and the integer control inputs $u \in \{-1, 0, 1\}$ describing the phase switches positions producing voltages $-V_{dc}/2, 0$ and $V_{dc}/2$ respectively. The system state is the current $i(t)$ through the load with dynamics being

$$\frac{\mathrm{d}i(t)}{\mathrm{d}t} = -\frac{R}{L} i(t) + \frac{1}{L} \frac{V_{dc}}{2} u(t). \quad (25)$$

R is the resistance and L the inductance of the load. The objective consists in tracking a perfect sinusoidal current $i^*(t)$ oscillating at frequency ω. Following the approach in [16] we can write the optimal current tracking problem in the form of a state regulation problem by adding two oscillating states. By applying the extension in Section VI-A

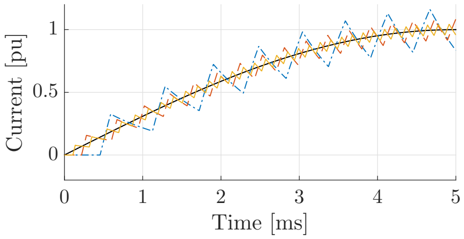

Fig. 2: Current Tracking Behavior for number of switchings $N = 25$ (dashed-dotted blue line), $N = 50$ (dashed red line) and $N = 100$ (solid yellow line).

we can add a further additional state to obtain the following autonomous switched dynamics:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \underbrace{\begin{bmatrix} -\frac{R}{L} & 0 & 0 & \frac{1}{L}\frac{V_{dc}}{2}u_i \\ 0 & 0 & -\omega & 0 \\ 0 & \omega & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A_i} x(t),\ y(t) = \underbrace{\begin{bmatrix} 1 & -1 & 0 & 0 \end{bmatrix}}_{C} x(t) \tag{26}$$

where matrices A_i are defined by the three possible control input variables $u_i \in \{-1, 0, 1\}$. The tracking problem minimizing the norm of the difference $i(t) - i^*(t)$ can be reconstructed by minimizing the quadratic function in (1) with matrix $Q = C^\top C$; see [16].

We simulate the power inverter with parameters $R = 2\,\Omega$, $L = 2\,\mathrm{mH}$, $V_{dc} = 5.2\,\mathrm{kV}$ and frequency $\omega = 2\pi 50\,{}^{\mathrm{rad}}/_{\mathrm{sec}}$. The system is represented in the per unit system (pu) with base current $I_B = \sqrt{2} I_R$ and rated current $I_R = 356\,\mathrm{A}$; see [15] for more details. In this way the magnitude of sinusoid i^* is 1. The initial state of the system is $\tilde{x}(0) = \begin{bmatrix} 0 & 0 & -1 & 1 \end{bmatrix}^\top$.

The number of switches is chosen to be 98 so that the switched dynamics iterate over the three modes 33 times. We choose a time window between 0 and $T = 5\,\mathrm{ms}$ equal to a quarter of the sinusoid. We choose this particular time window because it is normally used in approaches where the optimal switching patterns are computed offline [17] and then, according to the system symmetries, the optimal switchings and mode sequence are replicated over the whole period of the sinusoid.

Our approach converges in 18 iterations to locally optimal switchings up to precision 10^{-6} in $6.58\,\mathrm{sec}$ of which $74\,\mathrm{ms}$ and $6.50\,\mathrm{sec}$ are the times required by IPOPT and the function callbacks, respectively. The returned minimum is $J(\tau) = 4.53 \cdot 10^{-4}$.

Since this is a time-varying reference tracking problem, there exists no constant input that could track the sinusoid perfectly. Hence, we would like the number of switchings to be as high as possible to produce perfect sinusoids. A plot of this behavior for $N = 25, 50$ and 100 switching times is displayed in Figure 2.

We solved the switching time optimization problem with an increasing number of switchings N from 2 to 150 obtaining higher accuracy. Even though the computations increase, the number of iterations only grows logarithmically as shown in Figure 3.

VIII. Conclusion

In this paper we proposed a novel efficient second-order method to formulate and solve switching time optimization problems for switched linear systems with no integrations required. We derived efficiently computable expressions for the cost function, the gradient and the Hessian for which most of the computations are the same and can be greatly simplified. With our approach, the most expensive computations are reduced to individual independent parallelizable scalar exponentials. We provide two simulation examples showing the fast convergence of our method and its ability to handle more than a hundred switching times. Future improvements can be obtained by implementing a tailored solver to the particular structure of our formulation providing further reductions in the execution time.

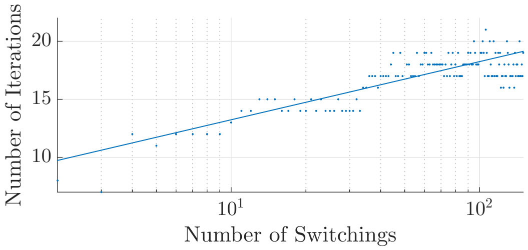

Fig. 3: Number of iterations required per number of switchings.

References

[1] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131, Apr. 2013.

[2] F. Zhu and P. J. Antsaklis, "Optimal control of hybrid switched systems: A brief survey," *Discrete Event Dynamic Systems*, vol. 25, no. 3, pp. 345–364, May 2014.

[3] C. Seatzu, D. Corona, A. Giua, and A. Bemporad, "Optimal Control of Continuous-Time Switched Affine Systems," *IEEE Trans. on Automatic Control*, vol. 51, no. 5, pp. 726–741, May 2006.

[4] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[5] M. Egerstedt, Y. Wardi, and H. Axelsson, "Transition-Time Optimization for Switched-Mode Dynamical Systems," *IEEE Trans. on Automatic Control*, vol. 51, no. 1, pp. 110–115, Jan. 2006.

[6] K. Flaßkamp, T. Murphey, and S. Ober-Blöbaum, "Discretized switching time optimization problems," in *2013 European Control Conference (ECC)*, Jul. 2013, pp. 3179–3184.

[7] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer Science & Business Media, 2006.

[8] E. R. Johnson and T. D. Murphey, "Second-Order Switching Time Optimization for Nonlinear Time-Varying Dynamic Systems," *IEEE Trans. on Automatic Control*, vol. 56, no. 8, pp. 1953–1957, Jul. 2011.

[9] T. M. Caldwell and T. D. Murphey, "Single Integration Optimization of Linear Time-Varying Switched Systems," *IEEE Trans. on Automatic Control*, vol. 57, no. 6, pp. 1592–1597, May 2012.

[10] B. Stellato, S. Ober-Blöbaum, and P. J. Goulart, "Second-Order Switching Time Optimization for Switched Dynamical Systems," *ArXiv e-prints*, Aug. 2016. [Online]. Available: http://arxiv.org/abs/1608.08597

[11] M. Gerdts, "A variable time transformation method for mixed-integer optimal control problems," *Optimal Control Applications and Methods*, vol. 27, no. 3, pp. 169–182, 2006.

[12] X. C. Ding, Y. Wardi, and M. Egerstedt, "On-Line Optimization of Switched-Mode Dynamical Systems," *IEEE Trans. on Automatic Control*, vol. 54, no. 9, pp. 2266–2271, Aug. 2009.

[13] R. H. Bartels and G. W. Stewart, "Solution of the matrix equation AX + XB = C," *Commun. ACM*, vol. 15, no. 9, pp. 820–826, Sep. 1972.

[14] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[15] T. Geyer and D. E. Quevedo, "Multistep Finite Control Set Model Predictive Control for Power Electronics," *IEEE Trans. on Power Electronics*, vol. 29, no. 12, pp. 6836–6846.

[16] B. Stellato, T. Geyer, and P. J. Goulart, "High-Speed Finite Control Set Model Predictive Control for Power Electronics," *IEEE Trans. on Power Electronics (To appear)*, 2016.

[17] J. Holtz and B. Beyer, "Fast current trajectory tracking control based on synchronous optimal pulsewidth modulation," *IEEE Trans. on Industry Applications*, vol. 31, no. 5, pp. 1110–1120, 1995.