Optimal Switching and Control of Nonlinear Switching Systems Using Approximate Dynamic Programming

Ali Heydari, *Member, IEEE*, and Sivasubramanya N. Balakrishnan, *Member, IEEE*

***Abstract*— The problem of optimal switching and control of switching systems with nonlinear subsystems is investigated in this paper. An approximate dynamic programming-based algorithm is proposed for learning the optimal cost-to-go function based on the switching instants and the initial conditions. The global optimal switching times for every selected initial condition are directly found through the minimization of the resulting function. Once the optimal switching times are calculated, the same neurocontroller is used to provide optimal control in a feedback form. Proof of convergence of the learning algorithm is presented. Two illustrative numerical examples are given to demonstrate the versatility and accuracy of the proposed technique.**

***Index Terms*— Approximate dynamic programming (ADP), neural networks (NNs), optimal switching.**

I. Introduction

FROM aerospace field to chemical processes, many examples exist in engineering that can be categorized as switching systems [1]–[5], in which subsystems with different dynamics exist and at each time instant, one of them is active. Hence, controlling these processes involves not only applying a stabilizing control to the system, but also making decisions on when to switch and what mode to switch to. Optimal switching and control of a switching system is a challenging problem and some methods have been developed to address the problem [6]–[18]. The main issue is to find optimal switching instants, and once they are found, the problem reduces to a conventional optimal control problem.

The methods developed so far for finding the optimal switching instants can be mainly divided to two groups: the first group comprises of nonlinear programming-based methods [6]–[13], in which through different schemes, the gradient of the cost with respect to the switching instants/points is calculated and then using a nonlinear optimization method, e.g., steepest descent, the switching instants/points are adjusted to find the local optimum. It should be noted that, in many existing papers, the sequence of active subsystems, called mode sequence, is selected *a priori* [6]–[12], and the problem reduces to finding the switching instants between the modes.

The second group includes studies that discretize the problem to deal with a finite number of options. Having a finite number of candidate switching time sequences, [14] uses a direct search to evaluate the cost function for different randomly selected switching time sequences and selects the best one in the sense of having less corresponding cost. In [15], the discretization of the state and input spaces is used for the calculation of the value function for optimal switching through dynamic programming.

In [16], genetic algorithm is used to find the optimal switching times. A neural network (NN) is used for solving the optimal switching problem for a prespecified initial condition in [17]. A hierarchical decomposition is proposed in [18], with the lower level being the time-driven dynamics and the higher level being the event-driven dynamics representing the mode switching.

All the cited methods numerically find the optimal switching time for a specific initial condition; each time the initial condition is changed, new computations are needed to find the new optimal switching instants. Recently, the authors of this paper proposed an NN-based scheme in [20] for optimal switching of systems with autonomous dynamics, i.e., where the subsystems do not admit control inputs. Switching with controlled subsystem, which is the subject of this paper, makes the problem much more complicated due to the intercoupling between the effect of switching to different modes and inputting different controls once each mode is active.

Within the last two decades, approximate dynamic programming (ADP) has shown a lot of promises in obtaining solutions to conventional optimal control problems with NN as the enabling structure [21]–[35]. ADP is usually carried out using a two-network synthesis called adaptive critics (ACs) [22]–[24]. In the heuristic dynamic programming (HDP) class with ACs, one network, called the critic network, inputs the states to the NN and outputs the optimal cost and another network, called the action network, outputs the control with states of the system as its inputs [24], [25]. In the dual heuristic programming formulation, while the action network remains the same as the HDP, the critic network outputs the costates with the current states as inputs [22], [26], and [27]. Note that the developments in [21]–[28] are for infinite-horizon

Manuscript received April 24, 2013; revised October 8, 2013; accepted October 20, 2013. Date of publication November 12, 2013; date of current version May 15, 2014. This work was supported by a grant from the National Science Foundation.

A. Heydari is with the Department of Mechanical Engineering, South Dakota School of Mines and Technology, Rapid City, SD 55701 USA (e-mail: ali.heydari@sdsmt.edu).

S. N. Balakrishnan is with the Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology, Rolla, MO 65401 USA (e-mail: bala@mst.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2288067

2162-237X © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

problems. The use of ADP for solving finite-horizon optimal control of conventional problems was considered in [30]–[34].

As for optimal control of switching problems, if the function that describes the optimal cost-to-go for every given switching time sequence is known explicitly, then the problem simplifies to minimization of the function with respect to the switching instants. However, even in the case of general linear subsystems with a quadratic cost function, this function is not available [7], [19]. The main contribution of this paper is developing an algorithm for switching problems that learns the optimal cost-to-go as a function of current state and the switching instants. An ADP-based scheme in an HDP form is used to train an NN to learn the nonlinear mapping between the optimal cost-to-go and the switching instants. Once this function is learned, finding the optimal switching times reduces to minimization of an analytical function. Furthermore, a second NN is trained along with to generate optimal control in a feedback form. Hence, once the optimal switching instants are calculated, one may use the control NN to generate the optimal control to be applied on the system.

As compared with available methods in the literature, the proposed technique has two advantages. They are: 1) the method developed in this paper gives global optimal switching instants versus local ones resulting from nonlinear programming-based methods and 2) the learned function gives the optimal cost based on the switching instants for a vast domain of initial conditions; hence, optimal switching times for different initial conditions can easily be calculated using the same trained NNs. Moreover, once the optimal switching instants are calculated, this method provides feedback optimal control, too. Convergence of the learning process is also provided.

The rest of this paper is organized as follows. The problem formulation is given in Section II, the proposed solution is described in Section III, numerical analysis is given in Section IV, and the conclusion is given in Section V.

II. Problem Formulation

A control-affine switching system can be represented by a set of M subsystems given by

$$\dot{x}(t) = F_{j(t)}(x(t)) + G_{j(t)}(x(t))\,u(t) \tag{1}$$

where $F_j : \mathbb{R}^n \to \mathbb{R}^n$ and $G_j : \mathbb{R}^n \to \mathbb{R}^{n\times m}$, $\forall j \in \mathcal{J} \equiv \{1, 2, \ldots, M\}$. Positive integers n and m denote the dimension of the state vector $x(t)$, and the control vector $u(t)$, respectively. Moreover, switching function $j : [t_0, t_f] \to \mathcal{J}$ returns the index of active subsystem at time $t \in [t_0, t_f]$, where t_0 and t_f are the initial and the final times, respectively. The technique developed in this paper requires that the given systems are in a control-affine form. If they are not control affine, some mathematical constructs (for example, defining a new control) are needed to convert the given system to a control-affine system [36]. The controller design process of switching systems includes not only selecting a history of control input $u(t)$, but also a switching function $j(t)$ that allows the operation of the system to switch between different subsystems. Following [7], the switching function can be given by a switching sequence as:

$$s = ((t_0, j_0), (t_1, j_1), \ldots, (t_K, j_K)) \tag{2}$$

where $t_0 \le t_1 \le \ldots \le t_K \le t_f$, $0 \le K < \infty$ and $j_k \in \mathcal{J}$, for $k = 1, 2, \ldots, K$. In this notation, (t_k, j_k) means that the system switches from subsystem j_{k-1} to j_k at time t_k, and K is the number of switching. Considering switching sequence s, switching function $j(t)$ is given by $j(t) = j_k$ where k is such that $t_k \le t < t_{k+1}$, $k \in \{0, 1, \ldots, K\}$, and t_{K+1} is defined as t_f.

The problem of optimal control of switching systems can be defined as determining a switching sequence s [which leads to a switching function $j(t)$] and a control $u(t)$, $t \in [t_0, t_f]$, such that the cost function given in the following is minimized:

$$J = \psi\left(x\left(t_f\right)\right) + \frac{1}{2}\int_{t_0}^{t_f}\left(Q(x(t)) + u(t)^T Ru(t)\right)dt. \tag{3}$$

Convex and smooth positive semidefinite functions $Q : \mathbb{R}^n \to \mathbb{R}$ and $\psi : \mathbb{R}^n \to \mathbb{R}$ penalize the states and positive definite matrix R penalizes the control effort. Following [6]–[12] in this paper, we freeze the order of the active subsystems and the number of switching K, i.e., the mode sequence, and work on finding the optimal switching times. Since the mode sequence is preselected, the unknowns in this problem will be the switching time sequence and the optimal control. A switching time sequence is given by

$$\tau = (t_0, t_1, t_2, \ldots, t_K) \tag{4}$$

where $t_0 \le t_1 \le \ldots \le t_K \le t_f$ and switching happens at t_ks, $k \in \{0, 1, \ldots, K\}$. Note that even for linear subsystems with quadratic cost functions, available methods in the literature give only a local optimal solution to this problem and only for a single set of initial conditions [6], [7].

Assumption 1: The functions representing the dynamics are smooth versus the state and control vectors x and u. In addition, there exists a control using which the resulting dynamics of the subsystems do not have finite escape times.

To have an idea of the implications of Assumption 1, it should be noted that if a function is (or equivalently all of its terms are) differentiable for infinite number of times, then the function is smooth. Therefore, for example, linear systems or systems with polynomial nonlinearities have smooth dynamics. As for the second condition, linear systems do not have a finite escape time [37]. Therefore, for example, nonlinear systems which are feedback linearizable, satisfy the desired condition of existence of a control using which the closed loop system cannot have a finite escape time. Moreover, nonlinear systems whose dynamics are globally Lipschitz do not have a finite escape time [37].

III. ADP Approach

An ADP framework is used in this paper as a solution technique to the optimal switching problem. First, we motivate the utilization of this approach for conventional optimal control problems and then proceed to using it for switching systems.

A. ACs for Conventional Optimal Control

To motivate the idea of using ADP for solving the switching problem, in this section, it is assumed that the switching time sequence is fixed and the unknown is the optimal control $u^*(t)$. This assumption reduces the switching problem to a conventional optimal control with a given cost function, wherein different subsystems are active at different given time periods [7]. In other words, the switching system simplifies to a system with time-varying dynamics. In the HDP scheme [23] with ADP, two NNs named actor and critic can be trained for approximating the optimal control and the optimal cost-to-go. Extending the idea of HDP to problems with finite-horizon cost function, the optimal control and the optimal cost-to-go are functions of the time-to-go (final time minus the current time) as well as the states [31]. Therefore, the actor and the critic are trained to capture the mapping between a given state and the time-to-go as inputs to the optimal control and the optimal cost-to-go as outputs for the actor and the critic, respectively.

Considering a fixed switching time sequence, switching system (1) simplifies to time-varying system

$$\dot{x}(t) = f(x(t), t) + g(x(t), t)\, u(t) \tag{5}$$

where $f(x,t) \equiv F_{j(t)}(x)$ and $g(x,t) \equiv G_{j(t)}(x)$. Note functions $f(x,t)$ and $g(x,t)$ are smooth with respect to x, but they can have discontinuity with respect to t at the fixed t_ks, due to the switching between the modes. Discretizing the system in (5) by selecting a sampling time Δt results in discrete-time dynamics of the subsystems

$$x_{k+1} = \bar{f}(x_k, k) + \bar{g}(x_k, k)\, u_k, \quad k = 0, 1, 2, \ldots, N \tag{6}$$

where k is the time index, $N = (t_f - t_0)/\Delta t$, $x_k = x(k\Delta t + t_0)$, and $\bar{f}(x,k) \equiv x + \Delta t f(x, k\Delta t + t_0)$, $\bar{g}(x,k) \equiv \Delta t g(x, k\Delta t + t_0)$ if forward-in-time Euler integration is used.

Let the cost function be discretized as

$$J = \psi(x_N) + \frac{1}{2}\sum_{k=0}^{N-1}\left(\bar{Q}(x_k) + u_k^T \bar{R} u_k\right) \tag{7}$$

where $\bar{Q}(x) \equiv \Delta t\, Q(x)$ and $\bar{R} \equiv \Delta t\, R$.

Remark 1: Dynamic programming [38], which is the backbone of the method developed here, gives solution to discrete-time problems. Therefore, the continuous-time problem is discretized. Moreover, the assumption that discrete-time system (6) is obtained through discretizing a continuous-time problem is used in convergence analysis of the developed algorithm in this paper. Note that almost all physical systems have subsystems with continuous-time dynamics; therefore, this assumption does not impose a limitation on the obtained results for such systems.

Let the optimal control and the optimal cost-to-go be denoted with superscripted "$*$" notation. The optimal cost-to-go at each instant may be denoted with $J_k^*(x_k)$ to emphasize its dependency on the current state, x_k, and on the remaining time, $N-k$. In other words

$$J_k^*(x_k) \equiv \psi(x_N) + \frac{1}{2}\sum_{k=k}^{N-1}\left(\bar{Q}(x_k^*) + u_k^* \bar{R} u_k^*\right). \tag{8}$$

The Bellman equation [38] provides an optimal solution to the problem of minimizing cost function (7) subjected to dynamics (6)

$$J_N^*(x_N) = \psi(x_N) \tag{9}$$

$$J_k^*(x_k) = \frac{1}{2}\left(\bar{Q}(x_k) + u_k^{*T} \bar{R} u_k^*\right) + J_{k+1}^*(x_{k+1}^*), \quad k = 0, 1, \ldots, N-1 \tag{10}$$

$$u_k^* = -\bar{R}^{-1}\bar{g}(x_k, k)^T \left.\frac{\partial J_{k+1}^*(x_{k+1})}{\partial x_{k+1}}\right|_{x_{k+1}^*}, \quad k = 0, 1, \ldots, N-1 \tag{11}$$

where $x_{k+1}^* = \bar{f}(x_k, k) + \bar{g}(x_k, k)\, u_k^*$. Gradient $\partial J_{k+1}^*(x_{k+1})/\partial x_{k+1}$ is formed as a column vector.

Denoting the approximated optimal cost-to-go and the approximated optimal control with $J_k(x_k)$ and $u_k(x_k)$, respectively, an iterative learning scheme can be derived from the Bellman equation for learning these unknowns for the fixed-final-time problem as [31]

$$J_N(x_N) = \psi(x_N) \tag{12}$$

$$u_k^{i+1}(x_k) = -\bar{R}^{-1}\bar{g}(x_k, k)^T \left.\frac{\partial J_{k+1}^*(x_{k+1})}{\partial x_{k+1}}\right|_{x_{k+1}^i}, \quad k = 0, 1, \ldots, N-1 \tag{13}$$

$$J_k(x_k) = \frac{1}{2}\left(\bar{Q}(x_k) + u_k(x_k)^T \bar{R} u_k(x_k)\right) + J_{k+1}(x_{k+1}), \quad k = 0, 1, \ldots, N-1. \tag{14}$$

Superscript i denotes the index of iteration. Moreover, in (13) one has $x_{k+1}^i \equiv \bar{f}(x_k, k) + \bar{g}(x_k, k) u_k^i(x_k)$, $x_{k+1} \equiv \bar{f}(x_k, k) + \bar{g}(x_k, k) u_k(x_k)$, and the converged value of u_k^i is denoted with u_k. Note that in a dual network AC scheme for finite-horizon optimal control, iteration takes place only in training the actor, as shown in (13). In other words, one starts with an initial guess on u_k^0, $k = 0, 1, \ldots, N-1$, and iterates using (13). Once the converged control value is obtained, optimal cost-to-go is calculated using (14), without any need for iteration. To circumvent the problem of curse dimensionality existing with the Bellman equation, two NNs are selected for learning the unknown functions, i.e., the control and the cost-to-go. By selecting linear in parameter/weight networks, the expressions for the actor (control) and the critic (cost) can be written as

$$u_k(x) = V_k^T \sigma(x), \quad k = 0, 1, \ldots, N-1 \tag{15}$$

$$J_k(x) = W_k^T \phi(x), \quad k = 0, 1, \ldots, N \tag{16}$$

where $V_k \in \mathbb{R}^{p \times m}$ and $W_k \in \mathbb{R}^q$ are the unknown weights of the actor and the critic networks at time step k, respectively. The selected linearly independent smooth basis functions are given by $\sigma : \mathbb{R}^n \to \mathbb{R}^p$ and $\phi : \mathbb{R}^n \to \mathbb{R}^q$ for p and q being positive integers denoting the number of neurons. The objective is using (12)–(14) to determine network weights V_k and W_k $\forall k$. Considering (12)–(14), unknowns $J_k(.)$ and $u_k(.)$ can be calculated in a backward-in-time fashion. In other words, using (12), one can calculate $J_N(.)$. Then, having $J_N(.)$ one can calculate $u_{N-1}(.)$ using the iterations given in (13). Having calculated $J_N(.)$ and $u_{N-1}(.)$, unknown $J_{N-1}(.)$ can be found using (14). Repeating this process from $k = N-1$

Algorithm 1 NN Training for Conventional Optimal Control Problems

Step 1: Find W_N such that $W_N^T\phi(x_N) \cong \psi(x_N)$ for different $x_N \in \Omega$ where Ω denotes a compact subset of $\mathbb{R}^n$ representing the domain of interest, which is a domain selected depending on the problem in which the state trajectory is supposed to lie.

Step 2: For $k = N-1$ to $k = 0$ repeat

{

Step 3: Set $i = 0$ and select a guess on V_k^0.

Step 4: Randomly select n many different state vectors $x_k^{(j)} \in \Omega$, $j \in \{1, 2, .., n\}$, for n being a large positive integer.

Step 5: Set $u_k^{(j)} = V_k^{i^T}\sigma\left(x_k^{(j)}\right)$, $j \in \{1, 2, .., n\}$.

Step 6: Set $x_{k+1}^{(j)} = \bar{f}\left(x_k^{(j)}, k\right) + \bar{g}\left(x_k^{(j)}, k\right)u_k^{(j)}$, $j \in \{1, 2, .., n\}$.

Step 7: Find V_k^{i+1} such that

$$V_k^{i+1^T}\sigma\left(x_k^{(j)}\right) \cong -\bar{R}^{-1}\bar{g}\left(x_k^{(j)}, k\right)^T\nabla\phi\left(x_{k+1}^{(j)}\right)^T W_{k+1}, \quad \forall j \in \{1, 2, .., n\}.$$

Step 8: Set $i = i+1$ and repeat Step 7, until $||V_k^{i+1} - V_k^i||$ converges to a number less than a small preset tolerance.

Step 9: Set $V_k = V_k^i$.

Step 10: Find W_k such that

$$W_k^T\phi\left(x_k^{(j)}\right) \cong \frac{1}{2}\bar{Q}\left(x_k^{(j)}\right) + \frac{1}{2}\sigma\left(x_k^{(j)}\right)^T V_k\bar{R}V_k^T\sigma\left(x_k^{(j)}\right) + W_{k+1}^T\phi\left(\bar{f}\left(x_k^{(j)}, k\right) + \bar{g}\left(x_k^{(j)}, k\right)V_k^T\sigma\left(x_k^{(j)}\right)\right), \quad \forall j \in \{1, 2, .., n\}.$$

to $k = 0$, all the unknowns can be calculated. This idea is used for training network weights V_k and W_k $\forall k$, as detailed in Algorithm 1. Note that the superscript on V denotes the iterationndex and $\nabla\phi(x) \equiv \partial\phi(x)/\partial x$ is formed as a column vector in the algorithm.

In Steps 1, 7, and 10 of Algorithm 1, the method of least squares, explained in Appendix A, can be used for finding the unknown weights in terms of the given parameters.

Remark 2: If the switching time sequence is fixed and given, optimal control of the time-varying system is found using the respective active subsystem at each time index k in the process of propagation of x_k to x_{k+1}, for use in the weight update equations of W_k and V_k. This step is the main difference between this process and the nonswitching AC-based finite-horizon optimal controllers given in [30] and [31].

Remark 3: Note that the number of iterations may vary for different time steps. The iteration index, i is reset to zero for the next time step.

Remark 4: One can modify the algorithm in the sense that at each iteration of Step 7, different random states $x_k^{(j)}$ being selected. For this purpose, one needs to repeat Steps 4–7 instead of only repeating Step 7 at each iteration. Note that as long as the number of samples n is large enough, selecting new samples at each iteration of Step 7 is not necessary.

Remark 5: Capability of uniform approximation of NNs [39], [40] shows that once the network is trained for a large enough number of samples from the domain of interest, denoted with n, the network is able to approximate the output for any new sample from the domain with a bounded approximation error. This error bound can be made arbitrarily small if the network is rich enough. For the linear in weight NN selected in this paper and the polynomial basis function used in the numerical examples, Weierstrass approximation theorem [41] proves a similar uniform approximation capability.

Remark 6: Considering the possible discontinuity of $\bar{g}(x_k, k)$ at the switching times (due to the switching between $G_j(.)$s,) and the presence of this term in the optimal control eq. (11), it is natural to have discontinuity in the optimal control history at the switching times, i.e., at t_ks. On the other hand, the uniform approximation property of NN holds for approximating a continuous function. However, since the NNs are selected with time-varying weights, and the switching times are fixed and given, the desired discontinuity can be formed using the discrete set of weights at different ks. To make it clearer, one should note that the control is a function of two variables, the time and the state. As long as function $u_k(x)$ for each given k is a continuous function versus x, it can be uniformly approximated using $V_k^T\sigma(x)$ by the given V_k. This desired continuity, however, can be extracted from the Bellman equation. Considering (9), smoothness of $J_N^*(.)$ follows from the smoothness of $\psi(.)$. Having smooth $J_N^*(.)$, $\bar{f}(., N-1)$, and $\bar{g}(., N-1)$ (Assumption 1), smoothness of $u_{N-1}^*(.)$ follows from (11). Moreover, the smoothness of $J_N^*(.)$ and $u_{N-1}^*(.)$ leads to the smoothness of $J_{N-1}^*(.)$ by (10). Repeating this argument from $k = N-1$ to $k = 0$, it is observed that the unknown functions $J_k^*(.)$ and $u_k^*(.)$ $\forall k$, are smooth, hence, continuous functions of x_k.

B. ACs for Switching Optimal Control

In this section, we modify the network structure and the training algorithm to find the optimal switching times along with the optimal control. To synthesize the ACs for switching systems, the critic network is trained to approximate the optimal cost-to-go for different switching time sequences. It should be made clear that both the optimal cost-to-go and control are functions of the selected switching time sequence, τ.

In this section, an approximation of $J_k^*(x_k, \tau)$ is learned as a function of x_k and τ and is denoted with $J_k(x_k, \tau)$. Considering time $k = 0$ the approximate optimal cost-to-go $J_0(x_0, \tau)$ for a set of initial conditions x_0 and a switching time sequence τ will be learnt first. Since the NN mapping functions are analytical, once $J_0(x_0, \tau)$ is learned, finding the global optimal τ for a given x_0 reduces to finding the global minimum of $J_0(x_0, \tau)$ with respect to τ. For example, if the problem involves only one switch, then the only unknown of the optimal switching sequence $\tau = (t_0, t_1)$ is t_1, and finding it is as simple as calculating the roots of the derivative of $J_0(x_0, \tau)$ with respect to t_1 and comparing the value of $J_0(x_0, \tau)$ at those roots along with the value at the boundary points to find the global minimum. As mentioned earlier,

even for linear systems with a quadratic cost function, function $J_0(x_0, \tau)$ is not known in a closed form, i.e., with respect to x_0 and τ [7], [19].

If the switching times are not fixed, the NNs fail to uniformly approximate $u_k^*(x_k, \tau)$, and hence $J_k^*(x_k, \tau)$, due to possible discontinuity of $u_k^*(x_k, \tau)$ versus k at the free switching times. To remedy this problem a transformation is used. The idea is to transform t to a new independent variable $\hat{t}$ such that the switching times are fixed in terms of $\hat{t}$ [7]. To motivate the basic idea, assume that the number of subsystems is two with one switching instant at t_1, at which the system switches from subsystem 1 to subsystem 2. Define a new independent variable $\hat{t} \in [0, 2]$ as

$$t = \begin{cases} t_0 + (t_1 - t_0)\,\hat{t} & \text{if } 0 \le \hat{t} < 1 \\ t_1 + (t_f - t_1)\,(\hat{t} - 1) & \text{if } 1 \le \hat{t} \le 2 \end{cases} \tag{17}$$

one has $t = t_0$ if $\hat{t} = 0$, $t = t_1$ if $\hat{t} = 1$, and $t = t_f$ if $\hat{t} = 2$. Using the new independent variable, the state equations given in (1) can be expressed as

$$x'(\hat{t}) = \begin{cases} (t_1 - t_0)\left(F_1\left(x\left(\hat{t}\right)\right) + G_1\left(x\left(\hat{t}\right)\right) u\left(\hat{t}\right)\right) & \text{if } 0 \le \hat{t} < 1 \\ (t_f - t_1)\left(F_2\left(x\left(\hat{t}\right)\right) + G_2\left(x\left(\hat{t}\right)\right) u\left(\hat{t}\right)\right) & \text{if } 1 \le \hat{t} \le 2 \end{cases} \tag{18}$$

where the prime notation, x' denotes the derivative of x with respect to $\hat{t}$. Cost function (3) then becomes

$$J = \psi(x(2)) + \frac{1}{2}(t_1 - t_0)\int_0^1 \left(Q\left(x\left(\hat{t}\right)\right) + u\left(\hat{t}\right)^T R u\left(\hat{t}\right)\right) d\hat{t} + \frac{1}{2}(t_f - t_1)\int_1^2 \left(Q\left(x\left(\hat{t}\right)\right) + u\left(\hat{t}\right)^T R u\left(\hat{t}\right)\right) d\hat{t}. \tag{19}$$

As can be seen, the benefit of the transformed time is the fact that the switching always happens at the fixed transformed time of $\hat{t} = 1$. Note that the actual switching time is still free and given by t_1. This feature provides the capability to have discontinuities in the history of the weights at $\hat{t} = 1$ to account for a possible discontinuity in control at $t = t_1$, as mentioned in Remark 6. For problems having more switches or more subsystems, e.g., the number of switching equals to K, this remedy can be extended where $\hat{t} \in [0, K+1]$ and the switches happen at $\hat{t} = 1, 2, \ldots, K$.

After performing the transformation of t to $\hat{t}$, one needs to discretize the resulting transformed dynamics and the transformed cost function [respectively, (18) and (19) if $K = 1$,] to end up with the discrete-time problem suitable for the ADP scheme. Let the transformed time $\hat{t}$ be discretized to N segments. Note that the initial time and the final time for $\hat{t}$ are 0 and $K + 1$, respectively, therefore $N = (K + 1)/\Delta\hat{t}$, where $\Delta\hat{t}$ denotes the sampling time for discretizing $\hat{t}$. The discretized dynamics read

$$x_{k+1} = \bar{f}(x_k, \tau, k) + \bar{g}(x_k, \tau, k)\, u_k, \quad k = 0, 1, 2, \ldots, N \tag{20}$$

where

$$\bar{f}(x, \tau, k) \equiv x + \Delta\hat{t}\left(t_{\not{k}+1} - t_{\not{k}}\right) F_{j_{\not{k}}}(x)$$

$\bar{g}(x, \tau, k) \equiv \Delta\hat{t}\left(t_{\not{k}+1} - t_{\not{k}}\right) G_{j_{\not{k}}}(x)$ and $\not{k}$ is such that $\not{k} \le k\Delta\hat{t} < \not{k} + 1$, $\not{k} \in \{0, 1, \ldots, K\}$. Note that functions $\bar{f}(x, \tau, k)$ and $\bar{g}(x, \tau, k)$ depend on the selected τ through the presence of $t_{\not{k}}$s in their definition. For example $\bar{f}(x, \tau, k)$ denotes the internal dynamics at state x, which is active at time k given the switching time sequence τ. The discretized cost function will be having time-varying penalizing terms as

$$J = \psi(x_N) + \frac{1}{2}\sum_{k=0}^{N-1}\left(\bar{Q}(x_k, \tau, k) + u_k^T \bar{R}(\tau, k) u_k\right) \tag{21}$$

where $\bar{Q}(x, \tau, k) \equiv \Delta\hat{t}\left(t_{\not{k}+1} - t_{\not{k}}\right) Q(x)$, and $\bar{R}(\tau, k) \equiv \Delta\hat{t}\left(t_{\not{k}+1} - t_{\not{k}}\right) R$ and $\not{k}$ is such that $\not{k} \le k\Delta\hat{t} < \not{k} + 1$, $\not{k} \in \{0, 1, \ldots, K\}$.

An interesting point in minimizing cost function (21) subjected to dynamics (20) is the fact that it is just a conventional optimal control problem with time-varying dynamics and cost function terms. The problem however, has free parameters $t_1, t_2, \ldots, t_K$ which form the switching time sequence τ. In other words, the switching time sequence in terms of the transformed time is given, but the switching time sequence in terms of the original time is still free and appears as scaling parameters $t_1, t_2, \ldots, t_K$ in (20) and (21). The mapping between the scaling parameters and the optimal control/cost-to-go can be easily learned using function approximation capability of NN. In the rest of this section, the ADP scheme is used for learning the optimal control and the optimal cost-to-go as functions of these scaling parameters. For this purpose, the following modified network structures are proposed:

$$u_k(x, \tau) = V_k^T \sigma(x, \tau), \quad k = 0, 1, \ldots, N-1 \tag{22}$$

$$J_k(x, \tau) = W_k^T \phi(x, \tau), \quad k = 0, 1, \ldots, N. \tag{23}$$

The inputs to the NNs are the current state and the switching time sequence τ, since the optimal control and optimal cost-to-go are dependent on these two parameters, as well as time. The dependency on time, however, is accounted for by using time-dependent weights. Let the switching time sequence τ be formed as a K-vector whose elements are the switching times. The new selected smooth basis functions are $\sigma : \mathbb{R}^n \times \mathbb{R}^K \to \mathbb{R}^p$ and $\phi : \mathbb{R}^n \times \mathbb{R}^K \to \mathbb{R}^q$.

Using these structures for the actor and critic networks along with dynamics (20) and cost function (21) in the iterative learning scheme (12)–(14), which were derived earlier based on Bellman equation, the weight update laws can now be determined. In this process it should be noted that functions and parameters $\bar{f}(x_k, k)$, $\bar{g}(x_k, k)$, $\bar{Q}(x_k)$, and $\bar{R}$ in (12)–(14) need to be replaced with $\bar{f}(x_k, \tau, k)$, $\bar{g}(x_k, \tau, k)$, $\bar{Q}(x_k, \tau, k)$, and $\bar{R}(\tau, k)$, respectively. Algorithm 2 gives the details of the resulting training/learning process to find V_k and W_k $\forall k$. In this algorithm, compact domain $\bar{\Omega}$ is defined as $\bar{\Omega} \equiv \left\{\tau = [t_1, t_2, \ldots, t_K]^T \in \mathbb{R}^K : t_0 \le t_1 \le \cdots \le t_K \le t_f\right\}$.

Remark 7: By comparing Algorithm 2 with Algorithm 1, it can be seen that the main modification is selecting random sets of parameters τ and training the networks to give optimal solution for every given τ.

The converged value, V_k, in Step 7 can be used in Step 10 and a least squares solution can be found for W_k once W_{k+1} and V_k are given, see Appendix A. The following theorem provides the sufficient condition for the convergence of iterative equation (24).

Algorithm 2 NN Training for Switching Optimal Control Problems

Step 1: Find W_N such that $W_N^T \phi(x_N, \tau) \cong \psi(x_N)$ for different random $x_N \in \Omega$ and different random $\tau \in \bar{\Omega}$.

Step 2: For $k = N-1$ to $k = 0$ repeat

{

Step 3: Set $i = 0$ and select a guess on V_k^0.

Step 4: Randomly select n many different state vectors $x_k^{(j)} \in \Omega$, and n many different switching time sequence $\tau^{(j)} \in \bar{\Omega}$, $j \in \{1, 2, .., n\}$, for n being a large positive integer.

Step 5: Set $u_k^{(j)} = V_k^{i^T} \sigma\left(x_k^{(j)}, \tau^{(j)}\right)$, $j \in \{1, 2, .., n\}$.

Step 6: Set $x_{k+1}^{(j)} = \bar{f}\left(x_k^{(j)}, \tau^{(j)}, k\right) + \bar{g}\left(x_k^{(j)}, \tau^{(j)}, k\right) u_k^{(j)}$.

Step 7: Find V_k^{i+1} such that

$$V_k^{i+1^T} \sigma\left(x_k^{(j)}, \tau^{(j)}\right) \cong -\bar{R}\left(\tau^{(j)}, k\right)^{-1} \bar{g}\left(x_k^{(j)}, \tau^{(j)}, k\right)^T \nabla\phi\left(x_{k+1}^{(j)}, \tau^{(j)}\right)^T \times W_{k+1}, \forall j \in \{1, 2, .., n\}. \quad (24)$$

Step 8: Set $i = i + 1$ and repeat Step 7, until $||V_k^{i+1} - V_k^i||$ converges to a number less than a small preset tolerance.

Step 9: Set $V_k = V_k^i$.

Step 10: Find W_k such that

$$W_k^T \phi\left(x_k^{(j)}, \tau^{(j)}\right) \cong \frac{1}{2}\bar{Q}\left(x_k^{(j)}, \tau^{(j)}, k\right) + \frac{1}{2}\sigma\left(x_k^{(j)}, \tau^{(j)}\right)^T V_k \bar{R}\left(\tau^{(j)}, k\right) V_k^T \sigma\left(x_k^{(j)}, \tau^{(j)}\right) + W_{k+1}^T \phi\left(\bar{f}\left(x_k^{(j)}, \tau^{(j)}, k\right) + \bar{g}\left(x_k^{(j)}, \tau^{(j)}, k\right) V_k^T \sigma\left(x_k^{(j)}, \tau^{(j)}\right), \tau^{(j)}\right), \forall j \in \{1, 2, .., n\}. \quad (25)$$

Theorem 1: The iterations given in Step 7 of Algorithm 2 converge for any selected initial guess on $V_k^0 \in \mathbb{R}^{p \times m}$ for $k = 0, 1, \ldots, N-1$, providing the sampling time selected for discretization of the continuous dynamics (1) is small enough.

The proof is given in Appendix B.

In Theorem 1, the role of the sampling time in discretization of the continuous system is emphasized. It is worthwhile to discuss this issue in detail. Substituting (22) and (23) in optimal control eq. (11) leads to

$$V_k^T \sigma\left(x_k^{(j)}, \tau^{(j)}\right) \cong -\bar{R}\left(\tau^{(j)}, k\right)^{-1} \bar{g}\left(x_k^{(j)}, \tau^{(j)}, k\right)^T \nabla\phi\left(\bar{f}\left(x_k^{(j)}, \tau^{(j)}, k\right) + \bar{g}\left(x_k^{(j)}, \tau^{(j)}, k\right) V_k^T \sigma\left(x_k^{(j)}, \tau^{(j)}\right), \tau^{(j)}\right)^T W_{k+1}, \forall j \in \{1, 2, .., n\} \quad (26)$$

which is the same as (24) except that V_k^{i+1} and V_k^i on both sides are replaced with V_k. Optimal weights V_k at each time instant k can be calculated by solving the nonlinear equation given in (26), without using the iteration given in (24). However, there is no analytical solution to the set of nonlinear equations (26) in general. Therefore, one needs to resort to numerical methods for solving the set of equations. Theorem 1 proves that for any given smooth dynamics and smooth basis functions, if the sampling time is small enough, the iterations given in (24) converge to the solution to the nonlinear equation (26). However, if the sampling time is fixed, certain conditions on the dynamics or the cost function weight terms need to hold in order for the iterations to converge. These conditions can be easily derived from the proof of Theorem 1.

Once the network weights converge, they solve Bellman equations (10) and (11) and satisfy the final condition (9). Therefore, the resulting cost-to-go and control are optimal. In practice, however, due to existence of NN approximation errors, the results will be an approximation of the optimal solution.

When $J_0(x_0, \tau)$ is learned using Algorithm 2, calculating the optimal switching time for any x_0 within the domain of interest reduces to a simple function minimization, i.e., minimizing $J_0(x_0, \tau)$ versus τ. Since the basis functions are smooth, they are of bounded variation [42], and hence have a finite number of minima in the compact set $\bar{\Omega}$ by definition [42], [43]. Therefore, the resulting $J_0(x_0, \tau) = W_0^T \phi(x_0, \tau)$ also will have a finite number of minima. Due to the smoothness of the function, the minima can be calculated using first derivative test, i.e., through finding the real stationary points, which are the real roots of $\partial J_0(x_0, \tau)/\partial\tau = 0$. Once all of the stationary points are calculated, comparing the value of the function at the stationary and boundary points, the global minimum can be determined.

Note that, for the result to be global optimum, two other conditions also need to hold, listed in the following.

1) Equations (12)–(14), and hence, (24) and (25) provide globally optimal input–target pairs for network training.
2) The NN training provides globally optimal weights for approximating the mapping between the given input–target pairs.

The proof that condition 1 holds follows from the proof of Theorem 1, once the conditions given in Theorem 1 hold. In other words, once it is proved that (24) is a contraction mapping, the uniqueness of fixed point V_k to the iterative equation (24) follows [44]. In other words, once the conditions given in Theorem 1 hold, ADP provides global optimal solution. Details of this result are beyond the scope of this paper and are given in [45]. Condition 2, also, holds as long as the method of least squares is used for calculating the NN weights. Because of the convexity of least squares problems [46], the concern of getting stuck in a local minimum does not exist.

As compared with [6]–[12], the advantages of the method presented here are twofold: 1) global optimal switching time sequence is obtained rather than local ones and 2) the method provides optimal switching times for any initial conditions as long as the resulting state trajectory lies within the domain on which the networks are trained, while the other studies are solutions for a selected initial condition. As shown

in this section, the approximated optimal control is given by (22). Once the network weights are trained and the optimal switching time sequence τ is calculated, (22) can be used for online calculation of the control in a feedback form.

Finally, it should be noted that the computational load of the method is mainly in the offline training of the NNs. However, as the order of the system or the number of the subsystems grows, the computational load will increase. If it becomes prohibitive, one should consider using reduced-order models or other techniques like decoupling the dynamics wherever possible.

IV. Numerical Analysis

A. Example 1

The first example is a switching nonlinear system with two subsystems and one switch. The subsystem dynamics are

$$\text{Subsystem 1}: \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_1^2 - x_2^2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad (27)$$

$$\text{Subsystem 2}: \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} u \quad (28)$$

where the elements of the state vector x are denoted by x_1 and x_2. As explained in the discussion after Assumption 1, it can be seen that the selected system satisfies the conditions given in the assumption. The reason is the fact that the subsystems are either linear or nonlinear with polynomial nonlinearity and also the nonlinear subsystem is feedback linearizable, for example using $u = -x_1^2 + x_2^2$. A quadratic cost function in the following form is selected:

$$J = x(t_f)^T S x(t_f) + \frac{1}{2}\int_0^{t_f} \left(x(t)^T Q x(t) + u(t)^T R u(t) \right) dt \quad (29)$$

where $t_f = 3$ s, $S = \text{diag}(10, 10)$, $Q = \text{diag}(1, 1)$, and $R = 1$. Once the independent variable t is transformed to $\hat{t}$, a sampling time of $\Delta\hat{t} = 0.002$ is used for discretization of the continuous dynamics. This sampling time was observed to lead to the convergence of the training algorithm. As long as the sampling time is selected small enough for the iterations given by (24) to converge, selecting any smaller sampling time was observed to have no noticeable effect on the performance of the resulting controller. However, using a smaller sampling time leads to needing more number of weights to train and store, hence, it is advisable to avoid selecting a sampling time which is too small.

The mode sequence is selected as (Subsystem1, Subsystem 2). Since there is only one switching, the switching time sequence simplifies to finding the switching time t_1. An important step in the design is the selection of the basis functions. Considering Remarks 5 and 6, the basis functions for the actor and critic are selected as polynomials made of different combinations of the two states and the switching time t_1. The selected basis functions for $\phi(.,.)$ are $t_1^a x_1^2$, $t_1^a x_2^2$, $t_1^a x_1 x_2$, $t_1^a x_1^3$, $t_1^a x_2^3$, $t_1^a x_1 x_2^2$, $t_1^a x_1^2 x_2$, $t_1^a x_1^4$, $t_1^a x_1^4$, $t_1^a x_1 x_2^3$, $t_1^a x_1^3 x_2$, and $t_1^a x_1^2 x_2^2$ for $a = 0, 1, 3, \ldots, 8$. The selections for $\sigma(.,.)$ are $t_1^a x_1$, $t_1^a x_2$, $t_1^a x_1^2$, $t_1^a x_2^2$, $t_1^a x_1 x_2$, $t_1^a x_1^3$, $t_1^a x_2^3$, $t_1^a x_1 x_2^2$, and $t_1^a x_1^2 x_2$ for $a = 0, 1, 3, \ldots, 8$. As with any linear in parameter networks, it is important to select suitable basis functions to allow the NNs for capturing the mappings accurately.

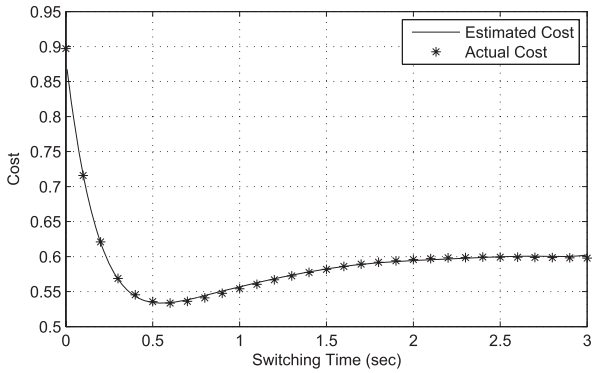


Fig. 1. Actual and estimated cost-to-go versus switching times for initial condition of $x_0 = [-1 \;\; 0.5]^T$, Example 1.

The domain of interest for the states is given by $\Omega = \{[x_1\, x_2]^T \colon x_1 \in [-1.25, 1.25], x_2 \in [-1.25, 1.25]\}$, and the domain of random switching times is $\bar{\Omega} = [0, 3]$. Five hundred random states and switching times from Ω and $\bar{\Omega}$, respectively, are used in the least squares process ($n = 500$). The iterative learning of V_k converged after less than three iterations. For initial conditions $x_0 = [-1 \;\; 0.5]^T$, function $J_0(x_0, t_1)$ which gives the optimal cost, is approximated by $W_0^T \phi(x_0, t_1)$ given in terms of t_1 as

$$\begin{aligned} J_0(x_0, t_1) = {} & +0.883 - 2.120 t_1 + 5.195 t_1^2 - 6.873 t_1^3 + 5.540 t_1^4 \\ & - 2.785 t_1^5 + 0.85 t_1^6 - 0.1437 t_1^7 + 0.010 t_1^8. \end{aligned} \quad (30)$$

Taking the derivative of $J_0(x_0, t_1)$ with respect to t_1 and setting it equal to zero gives three real roots of $t_1 = 0.55, 2.68$, and 2.85. Examining the roots and the boundary points shows that the global minimum of the cost happens at $t_1 = 0.55$ s with a resulting cost-to-go of 0.53.

To evaluate the preciseness of the estimated function $J_0(x_0, t_1)$, the optimal cost-to-go for different preselected switching times is calculated using Algorithm 1 for which separate pairs of networks are trained for 30 different switching times and the results are shown in Fig. 1 denoted by asterisks and compared with the approximated cost-to-go, i.e., function $J_0(x_0, t_1)$. It is seen that the NN accurately describes the cost function, and therefore optimal switch times. Once optimal t_1 is found, the actor network can be used for the selected t_1 to give the feedback optimal control for propagation of the states.

An important feature of the developed method is learning the cost function $J_0(x_0, t_1)$ for different initial conditions in Ω. To test this capability, the trained network is used for calculation of $J_0(x_0, t_1)$ for the new initial condition of $x_0 = [0 \;\; -1]^T$. Optimal cost (denoted by asterisks) and the outputs of the cost network for different t_1s are shown in Fig. 2. The results are quite identical.

B. Example 2

A more complex problem is presented in the following. Example 2 is a switching system with three linear subsystems

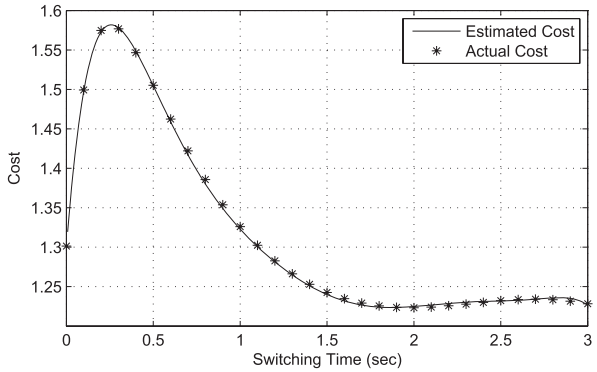


Fig. 2. Actual and estimated cost-to-go versus switching times for initial condition of $x_0 = [0 \quad -1]^T$, Example 1.

and two switches. The subsystem dynamics are:

$$\text{Subsystem 1}: \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ -1 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad (31)$$

$$\text{Subsystem 2}: \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} u \quad (32)$$

$$\text{Subsystem 3}: \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -.1 & 0 \\ 0 & -.1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.75 \end{bmatrix} u \quad (33)$$

where the elements of the state vector x are denoted by x_1 and x_2. Subsystem 1 is neither controllable nor stabilizable, subsystem 2 is controllable, and subsystem 3 is not controllable but it is stabilizable. Assumption 1 is satisfied because the subsystems are linear. The same quadratic cost function as in (29) is selected where $t_f = 3$ s, $S = \text{diag}(10, 10)$, $Q = \text{diag}(1, 1)$, $R = 1$, and once the independent variable t is transformed to $\hat{t}$, a sampling time of $\Delta\hat{t} = 0.002$ is used for discretization of the transformed continuous dynamics. Note that $0 \le \hat{t} \le 3$ because of two switches.

Switching sequence is composed of two switches here, with the mode sequence of (Subsystem1, Subsystem 2, Subsystem 3), and so the switching time sequence simplifies to two unknowns t_1 and t_2. Basis functions for the actor and critic are selected as polynomials made up of different combinations of the two states and the switching times t_1 and t_2. Assumed basis functions for $\phi(.,.)$ are $t_1^a x_1^2$, $t_1^a x_2^2$, $t_1^a x_1 x_2$ for $a = 0, 1, 2, \ldots, 8$ and also $t_1^a t_2^b x_1^2$, $t_1^a t_2^b x_2^2$, $t_1^a t_2^b x_1 x_2$ for $a = 1, 2, 3, 4$ and $b = 1, 2, 3, 4$ excluding $a = b = 4$. For $\sigma(.,.)$ the selections are $t_1^a x_1$, $t_1^a x_2$ for $a = 0, 1, 2, \ldots, 8$ and $t_1^a t_2^b x_1$, $t_1^a t_2^b x_2$ for $a = 1, 2, 3, 4$ and $b = 1, 2, 3, 4$ excluding $a = b = 4$. One could try other types of basis functions too.

The selected domain of interest for the states in training is the same as in Example 1. Moreover, $\bar{\Omega} = \{[t_1, t_2]^T \in \mathbb{R}^2 : 0 \le t_1 \le t_2 \le 3\}$. One thousand random states and random switching times from Ω and $\bar{\Omega}$, respectively, are used in the least squares process, i.e., $n = 1000$.

The evolution of the weights of the actor versus the training iterations is shown in Fig. 3. As shown in this figure, the iterations quickly converged in one iteration and the second iteration had a very slight effect in updating the weights. As can be observed from the weight histories shown in Fig. 4, weights evolve smoothly except at $\hat{t} = 1$ and $\hat{t} = 2$ where the switches happen. Note that in the weight history of the actor, there are jumps at the switching times; therefore, control values show jumps at the same instances, as explained in Remark 6. However, the weight history of the critic shows no jump, as expected, since it reflects an integrated value [see (29)].

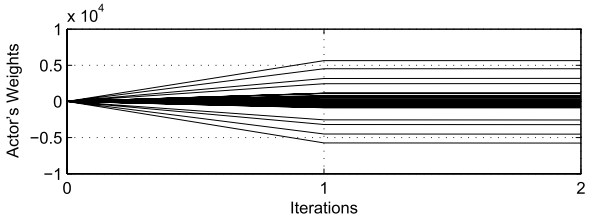


Fig. 3. Evolution of the weights of the actor during training iterations, Example 2.

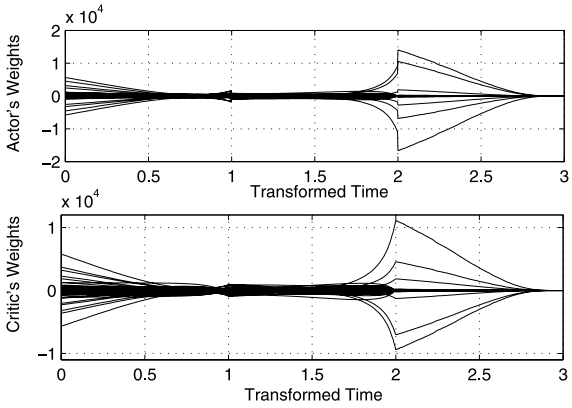


Fig. 4. Weights histories for the actor and critic versus transformed time, Example 2.

The optimal cost $J_0(x_0, t_1, t_2)$ after the weight training for initial condition of $x_0 = [1 \quad 1]^T$ is given by

$$\begin{aligned} J_0(x_0, t_1, t_2) = {} & 4.47 - 13.1t_1 + 9.46t_2 + 50.1t_1^2 - 3.00t_2^2 \\ & - 85.9t_1^3 - 6.84t_2^3 + 78.3t_1^4 + 7.01t_2^4 \\ & - 47.4t_1^5 - 2.95t_2^5 + 16.1t_1^6 + .638t_2^6 \\ & - 2.95t_1^7 - .07t_2^7 + .225t_1^8 + .004t_2^8 \\ & - 27.0t_1t_2 + 33.3t_1^2t_2 + 15.3t_1t_2^2 \\ & - 24.4t_1^2t_2^2 - 6.68t_1^3t_2 - 2.07t_1t_2^3 \\ & + 3.34t_1^3t_2^2 + 5.56t_1^2t_2^3 + 6.57t_1^4t_2 \\ & - .149t_1t_2^4 - 3.04t_1^4t_2^2 - .243t_1^2t_2^4 \\ & + .477t_1^4t_2^3 - .184t_1^3t_2^4. \end{aligned}$$

To evaluate the accuracy of the learned optimal cost-to-go $J_0(x_0, t_1, t_2)$, optimal costs for different preselected switching times t_1 varying from zero to three with the step size of 0.1 and t_2 varying from t_1 to three with the same step size, leading to 496 different switching time sequences, are calculated. The mean of the absolute value of the error between the optimal and approximated costs using $J_0(x_0, t_1, t_2)$ turned out to be 1.1% and the standard deviation of the absolute value of the error was 0.9%. These results show that the proposed technique leads to fairly accurate optimal cost over a wide region.

Fig. 5 shows $J_0(x_0, t_1, t_2)$ for different t_1 and t_2, where $t_2 \ge t_1$. As seen, $J_0(x_0, t_1, t_2)$ is not convex versus t_1 and t_2.

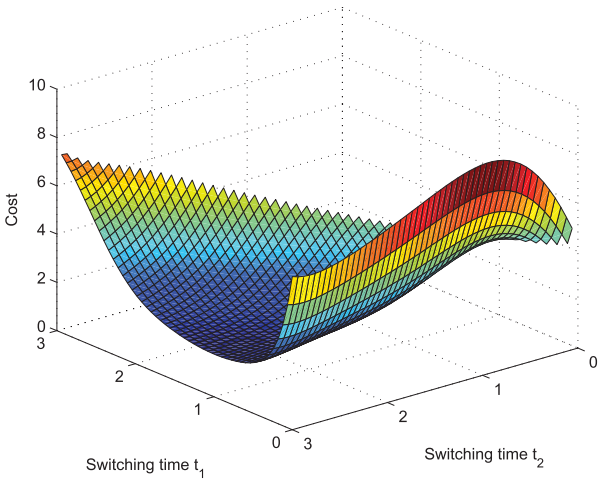


Fig. 5. Estimated cost-to-go versus switching times t_1 and t_2 for the selected initial condition, Example 2.

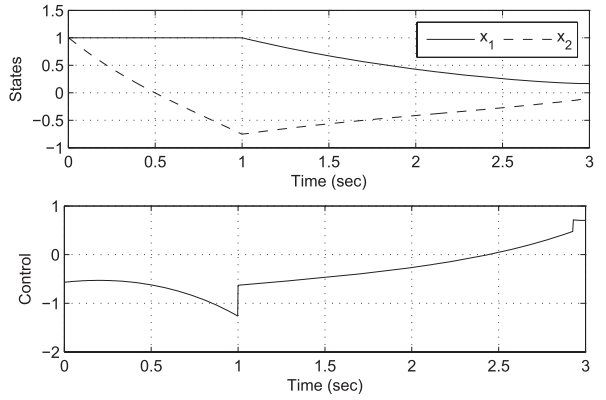


Fig. 6. State trajectories and control history for $t_1 = 1$ s and $t_2 = 2.93$ s, Example 2.

Note that this technique results in a polynomial expression of switching time and therefore, the global optimum can be found using the first derivative test whereas most of the existing methods assume a convex function, and therefore may end up with a local minimum. In other words, one can calculate all the stationary points of the resulting $J_0(x_0, t_1, t_2)$ and find the one which is the global minimum. For this problem, the globally optimal switching times turned out to be given by $t_1 = 1$ s, and $t_2 = 2.93$ s.

Fig. 6 shows the state trajectories and the control history from using the actor network and the calculated optimal switching times to control the system. As seen in the control history, there are two jumps in the control at the switching instants. Optimal switching time t_1 being equal to one means that the controller prefers to work with the uncontrollable Subsystem 1 for the first second and then switches to the controllable Subsystem 2. It is interesting to note how optimal switching has exploited the nature of the dynamics of different subsystems. Initial positive values for x_2 in Subsystem 2 leads to the growth of x_1, whereas in Subsystem 1 it not only does not alter x_1 but also the positive initial values of x_1 help the controller decrease x_2 without much control effort, hence, the controller has used Subsystem 1 until it controls x_2 to find some negative values and then switches to Subsystem 2 to force x_1 converge to the origin along with x_2. Lack of controllability of Subsystem 3 and its slow dynamics has caused the controller to prefer to work with the other two subsystems instead of Subsystem 3 for most of the simulation time.

V. Conclusion

An algorithm was developed for learning the optimal cost-to-go as a function of the current state and the switching time, for switching systems. Convergence of the given iterative weight update law was proved. Numerical analyses showed that the estimated function is accurate for the simulated switching systems with linear and nonlinear subsystems. These results show that the proposed method has a lot of potentials. The fact that once the networks are trained, global optimal switching times for different initial conditions can easily be obtained makes this method very versatile as compared with the other existing methods in the field of control of switching systems.

Appendix A

In Algorithms 1 and 2, in different steps, different weight update rules for the weights of the actor and critic networks, i.e., V_k and W_k, are given. The least squares method can be used for rewriting these equations such that V_k and W_k are explicitly given based on the known parameters. In this appendix, the process for finding such an equation for V_k from (24) is explained and one can easily find the corresponding equation for W_k.

To perform least squares for the weight update of V_k, n random states and n random switching sequence denoted by $x^{(j)}$ and $\tau^{(j)}$, respectively, where $1 \leq j \leq n$ are selected. Denoting the right-hand side of (24) resulting from each one pair of $x^{(j)}$ and $\tau^{(j)}$ with $y(x^{(j)}, \tau^{(j)})$, the objective is finding V_k such that it solves

$$\begin{cases} V_k^T \sigma\left(x^{(1)}, \tau^{(1)}\right) = y\left(x^{(1)}, \tau^{(1)}\right) \\ V_k^T \sigma\left(x^{(2)}, \tau^{(2)}\right) = y\left(x^{(2)}, \tau^{(2)}\right) \\ \vdots \\ V_k^T \sigma\left(x^{(n)}, \tau^{(n)}\right) = y\left(x^{(n)}, \tau^{(n)}\right) \end{cases} \tag{34}$$

Define

$$\boldsymbol{\sigma} \equiv \left[\sigma\left(x^{(1)}, \tau^{(1)}\right)\ \sigma\left(x^{(2)}, \tau^{(2)}\right) \cdots \sigma\left(x^{(n)}, \tau^{(n)}\right)\right]$$
$$\mathbf{y} \equiv \left[y\left(x^{(1)}, \tau^{(1)}\right)\ y\left(x^{(2)}, \tau^{(2)}\right) \cdots y\left(x^{(n)}, \tau^{(n)}\right)\right].$$

Using the method of least squares, solution to the system of linear equations (34) is given by

$$V_k = \left(\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right)^{-1} \boldsymbol{\sigma}\mathbf{y}^T. \tag{35}$$

Note that for the inverse of matrix $(\boldsymbol{\sigma}\boldsymbol{\sigma}^T)$, which is a $p \times p$ matrix, to exist, one needs the basis functions $\sigma(.,.)$ to be linearly independent and n to be greater than or equal to the number of the basis functions.

APPENDIX B

Proof of Theorem 1: The iteration performed on V_k^i, given in (24) and repeated here, is a successive approximation to find a fixed point of a function

$$\begin{aligned} &V_k^{i+1^T}\sigma\left(x_k^{(j)},\tau^{(j)}\right)\\ &\cong -\bar{R}\left(\tau^{(j)},k\right)^{-1}\bar{g}\left(x_k^{(j)},\tau^{(j)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(j)},\tau^{(j)},k\right)\right.\\ &\quad\left.+\bar{g}\left(x_k^{(j)},\tau^{(j)},k\right)V_k^{i^T}\sigma\left(x_k^{(j)},\tau^{(j)}\right),\tau^{(j)}\right)^T W_{k+1}. \end{aligned}$$

In order words, there exists function $F(.):\mathbb{R}^{p\times m}\to\mathbb{R}^{p\times m}$ such that (24) is of form

$$V_k^{i+1}=\mathcal{F}(V_k^i). \tag{36}$$

The claim of the theorem is proved if it can be shown that (36) is a contraction mapping [44]. Since $\mathbb{R}^{p\times m}$ with two-norm denoted by $||.||$ is a Banach space, iterations given by (36), regardless of initial V_k^0 converges to some $V_k=F(V_k)$ if there is a $0<\rho<1$ such that for every U_1 and U_2 in $\mathbb{R}^{p\times m}$, the following inequality holds [44]:

$$\|\mathcal{F}(U_1)-\mathcal{F}(U_2)\|\le\rho||U_1-U_2||. \tag{37}$$

Function $\mathcal{F}(.)$ can be formed by converting (24) to a least squares form performed in Appendix A. Rewriting (35), given in Appendix A, leads to the $\mathcal{F}(.)$ given at the bottom of this page. One has

$$\begin{aligned} \|\mathcal{F}(U_1)-\mathcal{F}(U_2)\|&\le\sqrt{n}\left\|\left(\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right)^{-1}\boldsymbol{\sigma}\right\|\\ &\times||\bar{R}\left(\tau^{(\ell)},k\right)^{-1}\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)\right.\\ &\quad\left.+\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)U_1^T\sigma\left(x_k^{(\ell)},\tau^{(\ell)}\right),\tau^{(\ell)}\right)^T W_{k+1}\\ &\quad-\bar{R}\left(\tau^{(\ell)},k\right)^{-1}\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)\right.\\ &\quad\left.+\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)U_2^T\sigma\left(x_k^{(\ell)},\tau^{(\ell)}\right),\tau^{(\ell)}\right)^T W_{k+1}|| \end{aligned} \tag{38}$$

where $\ell\in\{1,2,\ldots,n\}$ is such that

$$\begin{aligned} \ell=\underset{i\in\{1,2,\ldots,n\}}{\operatorname{argmax}}\;&||\bar{R}\left(\tau^{(i)},k\right)^{-1}\bar{g}\left(x_k^{(i)},\tau^{(i)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(i)},\tau^{(i)},k\right)\right.\\ &\left.+\bar{g}\left(x_k^{(i)},\tau^{(i)},k\right)U_1^T\sigma\left(x_k^{(i)},\tau^{(i)}\right),\tau^{(i)}\right)^T W_{k+1}\\ &-\bar{R}\left(\tau^{(i)},k\right)^{-1}\bar{g}\left(x_k^{(i)},\tau^{(i)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(i)},\tau^{(i)},k\right)\right.\\ &\left.+\bar{g}\left(x_k^{(i)},\tau^{(i)},k\right)U_2^T\sigma\left(x_k^{(i)},\tau^{(i)}\right),\tau^{(i)}\right)^T W_{k+1}||. \end{aligned}$$

In (38), the following norm inequality is used:

$$\left\|\begin{bmatrix}y_1\\y_2\\\vdots\\y_n\end{bmatrix}\right\|\le\sqrt{n}\,\|y_\ell\| \tag{39}$$

where y_is are real-valued row vectors and $\ell=\underset{i\in\{1,2,\ldots,n\}}{\operatorname{argmax}}\|y_i\|$.

Smoothness of $\phi(.,.)$ leads to the Lipschitz continuity of $\nabla\phi(.,.)$ on compact set Ω [47]. Therefore, there exists some positive real numbers ρ_ϕ, independent of τ, such that for every x_1 and x_2 in Ω and $\tau\in\bar{\Omega}$, one has $\|\nabla\phi(x_1,\tau)-\nabla\phi(x_2,\tau)\|\le\rho_\phi\|x_1-x_2\|$. Using this feature of $\nabla\phi(.,.)$, (38) can be written as

$$\begin{aligned} &\|\mathcal{F}(U_1)-\mathcal{F}(U_2)\|\\ &\le\rho_\phi\sqrt{n}\left\|\left(\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right)^{-1}\boldsymbol{\sigma}\right\|\left\|\bar{R}\left(\tau^{(\ell)},k\right)^{-1}\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)^T\right\|\\ &\quad\times\left\|\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)\right\|\left\|\sigma\left(x_k^{(\ell)},\tau^{(\ell)}\right)\right\|\|W_{k+1}\|\,\|(U_1^T-U_2^T)\|. \end{aligned} \tag{40}$$

By defining

$$\begin{aligned} \rho\equiv\rho_\phi\sqrt{n}&\left\|\left(\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right)^{-1}\boldsymbol{\sigma}\right\|\left\|\bar{R}\left(\tau^{(\ell)},k\right)^{-1}\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)^T\right\|\\ &\times\left\|\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)\right\|\left\|\sigma\left(x_k^{(\ell)},\tau^{(\ell)}\right)\right\|\|W_{k+1}\| \end{aligned} \tag{41}$$

one can select the sampling time $\Delta\hat{t}$ in discretization of the continuous dynamics (1) small enough such that the condition $0<\rho<1$ is satisfied, since a smaller $\Delta\hat{t}$, directly results in a smaller $\left\|\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)\right\|$ while the other terms including $\left\|\bar{R}\left(\tau^{(\ell)},k\right)^{-1}\bar{g}\left(x_k^{(\ell)},\tau^{(\ell)},k\right)^T\right\|$ are not affected. Note that smoothness, and hence continuity, of $G_j(.)$s and $\sigma(.,.)$ in their domains result in being bounded in the compact sets Ω and $\bar{\Omega}$ [42], therefore, the $x_k^{(\ell)}$ and $\tau^{(\ell)}$ dependent terms in (41) are upper bounded.

The expression given for the contraction mapping coefficient ρ in (41) involves $||W_{k+1}||$ also. It should be noted that W_{k+1} is already learned from the previous step in the algorithm, therefore, it is bounded. In other words, starting from $k=N-1$, one uses the successive approximation given by (24) and once V_k^i converges, it is used in (25) to calculate the bounded W_k. This process is repeated till $k=0$.

Note that if the selected sampling time $\Delta\hat{t}$ is not small enough, at some k, $0\le k\le N-1$, the respective ρ given in (41) does not satisfy condition $0<\rho<1$, therefore, V_k^i does not converge as $i\to\infty$. In that case, one may select a smaller sampling time and restart the algorithm, i.e., from

$$\mathcal{F}\left(V_k^i\right)\equiv\left(\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right)^{-1}\boldsymbol{\sigma}\begin{bmatrix}\left(-\bar{R}\left(\tau^{(1)},k\right)^{-1}\bar{g}\left(x_k^{(1)},\tau^{(1)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(1)},\tau^{(1)},k\right)+\bar{g}\left(x_k^{(1)},\tau^{(1)},k\right)V_k^{i^T}\sigma\left(x_k^{(1)},\tau^{(1)}\right),\tau^{(1)}\right)^T W_{k+1}\right)^T\\ \left(-\bar{R}\left(\tau^{(2)},k\right)^{-1}\bar{g}\left(x_k^{(2)},\tau^{(2)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(2)},\tau^{(2)},k\right)+\bar{g}\left(x_k^{(2)},\tau^{(2)},k\right)V_k^{i^T}\sigma\left(x_k^{(2)},\tau^{(2)}\right),\tau^{(2)}\right)^T W_{k+1}\right)^T\\ \vdots\\ \left(-\bar{R}\left(\tau^{(n)},k\right)^{-1}\bar{g}\left(x_k^{(n)},\tau^{(n)},k\right)^T\nabla\phi\left(\bar{f}\left(x_k^{(n)},\tau^{(n)},k\right)+\bar{g}\left(x_k^{(n)},\tau^{(n)},k\right)V_k^{i^T}\sigma\left(x_k^{(n)},\tau^{(n)}\right),\tau^{(n)}\right)^T W_{k+1}\right)^T\end{bmatrix}$$

$k = N - 1$ to calculate the weights corresponding to the smaller sampling time. Refining the sampling time leads to a change in W_{k+1} as well. However, it can be shown that as the sampling time becomes smaller, W_{k+1} remains bounded. This boundedness follows from looking at the definition of W_{k+1}, which is the weights for the network that approximates a discretized optimal cost-to-go. In other words:

$$W_{k+1}^T \phi(x_{k+1}, \tau) \cong \psi(x_N) + \frac{1}{2} \sum_{k=k+1}^{N-1} \bar{Q}(x_k, \tau, k) + u_k^T \bar{R}(\tau, k) u_k. \quad (42)$$

As the sampling times go to zero, the value of the discretized cost-to-go converges to the cost-to-go given by

$$J\left(x\left(\bar{t}\right), \bar{t}\right) = \psi\left(x\left(t_f\right)\right) + \frac{1}{2} \int_{\bar{t}}^{t_f} \left(Q(x(t)) + u(t)^T Ru(t)\right) dt \quad (43)$$

where $\bar{t}$ is the time corresponding to the transformed time $(k+1)\Delta\hat{t}$. On the other hand, since there exists a control using which the state trajectory cannot escape (Assumption 1) the trajectory will not escape using optimal control as well. Hence, the finite-horizon cost-to-go (43) will also be finite Note that the control history included in the integration in (43) corresponds to the already converged time steps, hence, they are (approximation) of the optimal control for the given τ. Therefore, as $\Delta\hat{t} \to 0$, the value of $W_{k+1}^T \phi(x_{k+1}, \tau)$ will be finite. Since the basis functions $\phi(x_{k+1}, \tau)$ are linearly independent, a finite $W_{k+1}^T \phi(x_{k+1}, \tau)$ leads to a finite W_{k+1}, as seen in the least squares operation described in Appendix A. Therefore, term $||W_{k+1}||$ existing in the expression for ρ in (41) remains bounded as the sampling time is refined. This completes the proof of convergence of V_k^i to V_k for $0 \leq k < N - 1$ using any initial guess on V_k^0, for any small enough sampling time. ■

REFERENCES

[1] M. Rinehart, M. Dahleh, D. Reed, and I. Kolmanovsky, "Suboptimal control of switched systems with an application to the DISC engine," *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 2, pp. 189–201, Mar. 2008.

[2] K. Benmansour, A. Benalia, M. Djemaï, and J. de Leon, "Hybrid control of a multicellular converter," *Nonlinear Anal., Hybrid Syst.*, vol. 1, no. 1, pp. 16–29, 2007.

[3] E. A. Hernandez-Vargas, R. H. Middleton, P. Colaneri, and F. Blanchini, "Dynamic optimization algorithms to mitigate HIV escape," in *Proc. IEEE Conf. Decision Control*, Dec. 2010, pp. 827–832.

[4] Z. Gong, C. Liu, E. Feng, L. Wang, and Y. Yu, "Modelling and optimization for a switched system in microbial fed-batch culture," *Appl. Math. Model.*, vol. 35, no. 7, pp. 3276–3284, 2011.

[5] M. Soler, A. Olivares, and E. Staffetti, "Framework for aircraft trajectory planning toward an efficient air traffic management," *J. Aircraft*, vol. 49, no. 1, pp. 985–991, 2012.

[6] X. Xu and P. J. Antsaklis, "Optimal control of switched systems via nonlinear optimization based on direct differentiations of value functions," *Int. J. Control*, vol. 75, no. 16, pp. 1406–1426, 2002.

[7] X. Xu and P. J. Antsaklis, "Optimal control of switched systems based on parameterization of the switching instants," *IEEE Trans. Autom. Control*, vol. 49, no. 1, pp. 2–16, Jan. 2004.

[8] M. Egerstedt, Y. Wardi, and H. Axelsson, "Transition-time optimization for switched-mode dynamical systems," *IEEE Trans. Autom. Control*, vol. 51, no. 1, pp. 110–115, Jan. 2006.

[9] H. Axelsson, M. Boccadoro, M. Egerstedt, P. Valigi, and Y. Wardi, "Optimal mode-switching for hybrid systems with varying initial states," *Nonlinear Anal., Hybrid Syst.*, vol. 2, no. 3, pp. 765–772, 2008.

[10] X. Ding, A. Schild, M. Egerstedt, and J. Lunze, "Real-time optimal feedback control of switched autonomous systems," in *Proc. IFAC Conf.*, 2009, pp. 108–113.

[11] M. Kamgarpoura and C. Tomlin, "On optimal control of non-autonomous switched systems with a fixed mode sequence," *Automatica*, vol. 48, no. 6, pp. 1177–1181, 2012.

[12] R. Zhao and S. Li, "Switched system optimal control based on parameterizations of the control vectors and switching instant," in *Proc. Chin. Control Decision Conf.*, 2011, pp. 3290–3294.

[13] J. Xu and Q. Chen, "Optimal control of switched hybrid systems," in *Proc. 8th Asian Control Conf.*, May 2011, pp. 1216–1220.

[14] R. Luus and Y. Chen, "Optimal switching control via direct search optimization," *Asian J. Control*, vol. 6, no. 2, pp. 302–306, 2004.

[15] M. Rungger and O. Stursberg, "A numerical method for hybrid optimal control based on dynamic programming," *Nonlinear Anal., Hybrid Syst.*, vol. 5, no. 2, pp. 254–274, 2011.

[16] M. Sakly, A. Sakly, N. Majdoub, and M. Benrejeb, "Optimization of switching instants for optimal control of linear switched systems based on genetic algorithms," in *Proc. IFAC Int. Conf. Intell. Control Syst. Signal Process.*, 2009, pp. 249–253.

[17] R. Long, J. Fu, and L. Zhang, "Optimal control of switched system based on neural network optimization," in *Proc. Int. Conf. Intell. Comput.*, 2008, pp. 799–806.

[18] K. Gokbayrak and C. G. Cassandras, "A hierarchical decomposition method for optimal control of hybrid systems," in *Proc. Decision Control Conf.*, vol. 2. 2000, pp. 1816–1821.

[19] W. Zhang, J. Hu, and A. Abate, "On the value functions of the discrete-time switched LQR problem," *IEEE Trans. Autom. Control*, vol. 54, no. 11, pp. 2669–2674, Nov. 2009.

[20] A. Heydari and S. N. Balakrishnan, "Optimal multi-therapeutic HIV treatment using a global optimal switching scheme," *Appl. Math. Comput.*, vol. 219, no. 14, pp. 7872–7881, 2013.

[21] P. J. Werbos, *Approximate Dynamic Programming for Real-Time Control and Neural Modeling*, D. A. White and D. A. Sofge, Eds. Brentwood, U.K.: Multiscience Press, 1992.

[22] S. N. Balakrishnan and V. Biega, "Adaptive-critic based neural networks for aircraft optimal control," *J. Guid., Control Dyn.*, vol. 19, no. 4, pp. 893–898, 1996.

[23] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 997–1007, Sep. 1997.

[24] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern.-Part B*, vol. 38, no. 4, pp. 943–949, Aug. 2008.

[25] T. Dierks and S. Jagannathan, "Online optimal control of affine nonlinear discrete-time systems with unknown internal dynamics by using time-based policy update," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1118–1129, Jul. 2012.

[26] S. Ferrari and R. F. Stengel, "Online adaptive critic flight control," *J. Guid., Control Dyn.*, vol. 27, no. 5, pp. 777–786, 2004.

[27] G. K. Venayagamoorthy, R. G. Harley, and D. C. Wunsch, "Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 764–773, May 2002.

[28] R. Padhi, N. Unnikrishnan, X. Wang, and S. N. Balakrishnan, "A single network adaptive critic (SNAC) architecture for optimal control synthesis for a class of nonlinear systems," *Neural Netw.*, vol. 19, no. 10, pp. 1648–1660, 2006.

[29] J. Ding and S. N. Balakrishnan, "An online nonlinear optimal controller synthesis for aircraft with model uncertainties," in *Proc. AIAA Guid., Navigat., Control Conf.*, 2010, pp. 1–5.

[30] D. Han and S. N. Balakrishnan, "State-constrained agile missile control with adaptive-critic-based neural networks," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 4, pp. 481–489, Jul. 2002.

[31] A. Heydari and S. N. Balakrishnan, "Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 1, pp. 145–157, Jan. 2013.

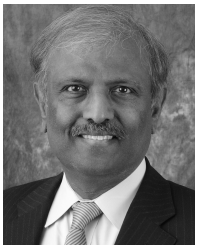
[32] F. Wang, N. Jin, D. Liu, and Q. Wei, "Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with ε -error bound," *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 24–36, Aug. 2011.

[33] R. Song and H. Zhang, "The finite-horizon optimal control for a class of time-delay affine nonlinear system," *Neural Comput. Appl.*, vol. 22, no. 2, pp. 229–235, 2011.
[34] Q. Wei and D. Liu, "An iterative ε-optimal control scheme for a class of discrete-time nonlinear systems with unfixed initial state," *Neural Netw.*, vol. 32, pp. 236–244, Aug. 2012.
[35] X. Xu, Z. Hou, C. Lian, and H. He, "Online learning control using adaptive critic designs with sparse kernel machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 762–775, May 2013.
[36] S. H. Lane and R. F. Stengel, "Flight control design using nonlinear inverse dynamics," in *Proc. Amer. Control Conf.*, 1986, pp. 587–596.
[37] S. Sastry, *Nonlinear Systems, Analysis, Stability, and Control*. New York, NY, USA: Springer-Verlag, 1999, pp. 10–12.
[38] D. E. Kirk, *Optimal Control Theory: An Introduction*. New York, NY, USA: Dover Publications, 2004.
[39] K. Homik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.
[40] J. G. Attali and G. Pages, "Approximations of functions by a multilayer perceptron: A new approach," *Neural Netw.*, vol. 10, no. 6, pp. 1069–1081, 1997.
[41] M. Stone and P. Goldbart, *Mathematics for Physics—A Guided Tour for Graduate Students*, Cambridge U.K.: Cambridge Univ. Press, 2009.
[42] W. Rudin, *Principles of Mathematical Analysis*, New York, NY, USA: McGraw-Hill, 1964.
[43] W. F. Trench, (2012). *Introduction to Real Analysis*, [Online]. Available: http://ramanujan.math.trinity.edu/wtrench/texts/trench_real_analysis.pdf
[44] H. Khalil, *Nonlinear Systems*, 3rd ed., Upper Saddle River, NJ, Prentice-Hall, 2002.
[45] A. Heydari and S. N. Balakrishnan, "Approximate dynamic programming; Global or local optimal solution," under review in Applied Soft Computing.
[46] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, U.K.: Cambridge Univ. Press, 2009, pp. 4–7.
[47] J. E. Marsden, T. Ratiu, and R. Abraham, *Manifolds, Tensor Analysis, and Applications*, 3rd ed. New York, NY, USA: Springer-Verlag, 2001.

Ali Heydari (S'09–M'14) received the Ph.D. degree in mechanical engineering from the Missouri University of Science and Technology, Rolla, MO, USA, in 2013.

He is currently an Assistant Professor of mechanical engineering with the South Dakota School of Mines and Technology, Rapid City, SD, USA. His current research interests include optimal control, nonlinear control, approximate dynamic programming, and control of hybrid and switching systems.

Dr. Heydari was a recipient of the Outstanding M.Sc. Thesis Award from the Iranian Aerospace Society, the Best Student Paper Runner-Up Award from the AIAA Guidance, Navigation and Control Conference, and the Outstanding Graduate Teaching Award from the Academy of Mechanical and Aerospace Engineers at Missouri S&T.



Sivasubramanya N. Balakrishnan (M'05) received the Ph.D. degree in aerospace engineering from the University of Texas, Austin, TX, USA.

He is currently a Curators' Professor of aerospace engineering with the Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology, Rolla, MO, USA. His current research interests include neural networks, optimal control, and control of large-scale and impulse systems. His papers from the development of techniques in these areas include applications to missiles, spacecraft, aircraft, robotics, temperature, and animal population control.