Applied Soft Computing 24 (2014) 291–303

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Global optimality of approximate dynamic programming and its use in non-convex function minimization

Ali Heydari [a,*], S.N. Balakrishnan [b]

[a] Mechanical Engineering, South Dakota School of Mines and Technology, United States
[b] Department of Aerospace Engineering, Missouri University of Science and Technology, United States

ARTICLE INFO

Article history:
Received 24 July 2013
Received in revised form 4 February 2014
Accepted 6 July 2014
Available online 18 July 2014

Keywords:
Approximate dynamic programming
Fixed final time optimal control
Neural networks
Non-convex function minimization

ABSTRACT

This study investigates the global optimality of approximate dynamics programming (ADP) based solutions using neural networks for optimal control problems with fixed final time. Issues including whether or not the cost function terms and the system dynamics need to be convex functions with respect to their respective inputs are discussed and sufficient conditions for global optimality of the result are derived. Next, a new idea is presented to use ADP with neural networks for optimization of non-convex smooth functions. It is shown that any initial guess leads to direct movement toward the proximity of the global optimum of the function. This behavior is in contrast with gradient based optimization methods in which the movement is guided by the shape of the local level curves. Illustrative examples are provided with single and multi-variable functions that demonstrate the potential of the proposed method.

Introduction

In the last two decades, approximate dynamics programming (ADP) has been shown to have a great promise in solving optimal control problems with neural networks (NN) [1–15]. In the ADP framework, the solutions are obtained using a two-network synthesis called adaptive critics (ACs) [2–4]. In the heuristic dynamic programming (HDP) approach with ACs, one network, called the 'critic' network, maps the input states to output the cost-to-go and another network, called the 'action' network, outputs the control with states of the system as its inputs [4,5]. In the dual heuristic programming (DHP) formulation, the action network remains the same as in the HDP, however, the critic network outputs the costates with the current states as inputs [2,6,7]. The computationally effective single network adaptive critics (SNAC) architecture consists of one network only. In [8], the action network was eliminated in a DHP type formulation with control being calculated from the costate values. Similarly, the J-SNAC [9] eliminates the need for the action network in an HDP scheme. Note that the developments in [1–9] are for *infinite-horizon* problems.

The use of ADP for solving *finite-horizon* optimal control problems was considered in [10–15]. Authors of [10] developed a time-varying neurocontroller for solving a scalar problem with state constraints. In [11] a single NN with a single set of weights was proposed which takes the time-to-go as an *input* along with the states and generates the fixed-final-time optimal control for discrete-time nonlinear multi-variable systems. An HDP based scheme for optimal control problems with *soft* or *hard* terminal constraints was presented in [12]. Finite-horizon problems with *unspecified* terminal times were considered in [13–15]. For an extensive literature on adaptive critic based problems, the reader is referred to [16] and the references in the chapters.

Despite much published literature on adaptive critics, there still exists an open question about the nature of optimality of the adaptive critic based results. Are they locally or globally optimal? A major contribution of this study is in proving that the AC based solutions are globally optimal subject to the assumed basis functions. To help with the development of the proof, the ADP based algorithm for solving fixed-final-time problems developed in [11,12] is revisited first. After describing the algorithm, a novel analysis of global optimality of the result is presented. It is shown that with any cost function with a quadratic control penalizing term, the resulting cost-to-go function (sometimes called value function) will be convex versus the control at the current time if the sampling time used for discretization of the original continuous-time system is small enough, and hence, the first order necessary optimality condition [17] will lead to the *global* optimal control. The second major contribution of this paper is in showing that the ADP can be used for functional optimization, specifically, optimization of non-convex functions. Finally, through numerical simulations, two examples with varying complexities are presented and the performance of

* Corresponding author. Tel.: +1 5732018645.
E-mail addresses: ali.heydari@sdsmt.edu (A. Heydari), bala@mst.edu (S.N. Balakrishnan).

http://dx.doi.org/10.1016/j.asoc.2014.07.003
1568-4946/

the proposed method is investigated. It is shown that despite the gradient based methods, selecting any initial guess on the minimum and updating the guess using the control resulting from the actor, the states will move *directly* toward the global minimum, passing any possible local minimum in the path.

The rest of this paper is organized as follows: The problem formulation is given in Section 'Problem formulation'. The ADP-based solution is discussed in Section 'Approximate dynamics programming based solution'. The supporting theorems and analyses are presented in Section 'Supporting theorems and analyses'. The use of the method in static function optimization is discussed in Section 'Non-convex function optimization', and the conclusions are given in Section 'Conclusions'.

Problem formulation

Let the control-affine dynamics of the system be given by

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t) \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$, and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$. The state and control vectors are denoted with $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$, respectively, where positive integers n and m denote the dimensions of the respective vectors. The selected cost function, J is fairly general but quadratic in control:

$$J = \psi(x(t_f)) + \int_{t_0}^{t_f} (Q(x(t)) + u(t)^T R u(t))dt \tag{2}$$

where positive semi-definite smooth 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. The initial and final time are denoted with t_0 and t_f, respectively. Discretizing the time horizon to N time steps using sampling time $\triangle t$ leads to the discrete-time dynamics and cost function as

$$x_{k+1} = \bar{f}(x_k) + \bar{g}(x_k)u_k, \quad k \in K \tag{3}$$

$$J = \psi(x_N) + \sum_{k=0}^{N-1} (\bar{Q}(x_k) + u_k^T \bar{R} u_k) \tag{4}$$

where $K \equiv \{0, 1, 2, \ldots, N-1\}$, $N \equiv (t_f - t_0/\Delta t)$, $x_k \equiv x(k\Delta t + t_0)$, $\bar{f}(x) \equiv x + \Delta t f(x)$, $\bar{g}(x) \equiv \Delta t g(x)$, $\bar{Q}(x) \equiv \Delta t Q(x)$, and $\bar{R} \equiv \Delta t R$, if Euler integration is used. The problem is defined as finding a control history $u_k \in \mathbb{R}^m$, $\forall k \in K$, such that cost function (4) is minimized subject to the dynamics given in (3).

Assumption 1. The dynamics of the system do not have finite escape times. Also, the functions $f(x)$ and $g(x)$ are smooth in x.

Remark 1. In order to use ADP, the continuous-time problem is discretized. Moreover, the assumption that discrete-time system (3) is obtained through discretizing a continuous-time problem is utilized in convergence analysis of the algorithm.

Approximate dynamics programming based solution

In this section, an ADP scheme called AC is used for solving the fixed-final-time optimal control problem in terms of the network weights and selected basis functions. The method is adopted from [11,12]. In this scheme, two networks called critic and actor are trained to approximate the optimal cost-to-go and the optimal control, respectively. It should be noted that the optimal cost-to-go, which represents the incurred cost if optimal decisions are made from the current time to the final time, at each instant is a function of the current state, x_k, and the current time, k. Denoting the cost-to-go with $J_k^*(x_k)$, one has

$$J_k^*(x_k) = \psi(x_N) + \sum_{\bar{k}=k}^{N=1} (\bar{Q}(x_{\bar{k}}) + u_{\bar{k}}^{*T} \bar{R} u_{\bar{k}}^{*T}) \tag{5}$$

where u_k^* denotes the optimal control at time k given the current state x_k. The solution to the problem is given by the Bellman equation [18] as

$$J_N^*(x_N) = \psi(x_N) \tag{6}$$

$$J_k^*(x_k) = \bar{Q}(x_k) + u_k^{*T} \bar{R} u_k^* + J_{k+1}^*(x_{k+1}^*), \quad k \in K \tag{7}$$

$$u_k^* = arg\min_{u_k \in \mathbb{R}^m} (\bar{Q}(x_k) + u_k^T \bar{R} u_k + J_{k+1}^*(\bar{f}(x_k) + \bar{g}(x_k)u_k)), \quad k \in K \tag{8}$$

where $x_{k+1}^* (\bar{f}(x_k) + \bar{g}(x_k)u_k^*)$. Applying the first order optimality condition [4,17], Eq. (8) leads to

$$u_k^* = -\frac{1}{2} \bar{R}^{-1} \bar{g}(x_k)^T \nabla J_{k+1}^*(x_{k+1}^*), \quad k \in K \tag{9}$$

where $\nabla J_{k+1}^*(x_{k+1}^*)$ denotes gradient $(\partial J_{k+1}^*(x_{k+1})/\partial x_{k+1})$ evaluated at x_{k+1}^* and formed as a column vector.

An iterative learning scheme can be derived from Bellman equation for finding the solution to the fixed-final-time problem once Eq. (9) is replaced with [12]

$$u_k^{i+1} = -\frac{1}{2} \bar{R}^{-1} \bar{g}(x_k)^T \nabla J_{k+1}^*(\bar{f}(x_k) + \bar{g}(x_k)u_k^i), \quad k \in K \tag{10}$$

Superscript i denotes the index of iteration which starts with an initial guess on u_k^0, $k \in K$. The converged value of u_k^i in (10) is denoted with u_k^* and used in (7). Note that in a dual network AC scheme for finite horizon optimal control, 'iterations' takes place in the training of the actor, as seen in (10). Once state-control relationship is learned, the optimal cost-to-go is obtained in a 'one-shot' process as given in (7).

Denoting the approximated optimal cost-to-go and the approximated optimal control in a feedback form with $J_k(x_k)$ and $u_k(x_k)$, respectively, and selecting linear in the weights NNs, the expressions for the actor (control) and the critic (cost), can be written as

$$u_k(x) = V_k^T \sigma(x), \; k \in K \tag{11}$$

$$J_k(x) = W_k^T \phi(x), \; k \in K \cup \{N\} \tag{12}$$

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, and the selected smooth basis functions are given by $\sigma : \mathbb{R}^n \to \mathbb{R}^p$ and $\phi : \mathbb{R}^n \to \mathbb{R}^q$ where p and q denote the number of neurons. The idea is using Eqs. (6), (7), and (10) to find the NN weights. Note that, once $J_{k+1}^*(\cdot)$ is known one can use (10) to find $u_k^*(.)$ and then (7) gives $J_k^*(.)$. Therefore, starting with (6) to find $J_N^*(.)$, all the unknowns can be calculated in a backward in time fashion, i.e., from $k = N-1$ to $k = 0$. The learning process for calculating weights V_k and W_k, $\forall k$ is detailed through Algorithm 1. Note that $\nabla\phi(x) \equiv (\partial\phi(x)/\partial x)$ is a column vector.