Proceedings of the American Control Conference
San Diego, California • June 1999

# Error bounds in nonlinear control design via Approximate Policy Iteration

C. I. Boussios, M. A. Dahleh and J. N. Tsitsiklis
Massachusetts Institute of Technology
Cambridge, MA 02139

## Abstract

We consider the optimal nonlinear control problem and evaluate a computational design procedure which produces suboptimal controllers (policies), approximate policy iteration. The method uses an approximation of the cost (cost-to-go) function of a given closed loop system in order to produce an updated controller of, hopefully, improved performance. We develop bounds on the approximation error such that the resulting controllers are stabilizing and bounds on the approximation error such that the resulting controllers are of improved performance. [1]

## 1 Introduction

We consider the problem of control design for nonlinear dynamic systems with Euclidean state and control input spaces. We assume a permormance index (quadratic cost function), and a given stabilizing controller. The objective is designing a state feedback controller, or policy[2], of improved performance.

Given a dynamic model in continuous time and the cost function, we generate a discrete time model for simulating the original, continuous time system. Advantages of an optimal control approach include generality in comparison to several other nonlinear control paradigms, addressing the issue of performance and possessing some desirable robustness properties [5]. Although analytical solution of the optimal control problem is generally difficult, the solution is characterized by Bellman's equation [1]. The latter gives rise to several algorithms for solving the problem which draw from the idea behind Bellman's equation and are collectively referred to as Dynamic Programming [1]. Unfortunately, dynamic programming algorithms suffer from the *curse of dimensionality*, that is, intractability of exact computation due to the fact that the optimal control has to be computed at each point of the state space.

Several approaches for nonlinear control using **Approximate Dynamic Programming** algorithms [2] focus on computational feasibility. In this paper, we evaluate *Approximate Policy Iteration*, one of the main Approximate Dynamic Programming algorithms, as a nonlinear control design tool. It is an iterative method that produces a sequence of policies based on off-line simulation of the dynamic system. It has been successfully used in several applications [8, 9], mainly for problems with discrete state spaces. In Section 2, we define the nonlinear control problem and present the algorithm. In Section 3 we develop conditions (bounds) on the allowed approximation error under which the resulting closed loop system, controlled by the iterate policy, is exponentially stable, and with improved performance. The development of error bounds offers insight into the design method. By demonstrating that the allowed approximation error bounds are "large", as argued in Sections 3.3 and 3.5, we aim at establishing the method's credibility and control engineers' confidence in it, despite the inevitability of approximation. **Remark:** For the case where no stabilizing controller is known for a system, a semi-globally stabilizing controller may be obtained via a proper modification of the algorithm (Chapter 4 of [3]).

## 2 Problem formulation and the design algorithm

### 2.1 Problem definition

We consider a continuous time nonlinear dynamic system of the form

$$\dot{x} = f(x) + G(x)u, \qquad (1)$$

where $f : \mathbf{R}^n \to \mathbf{R}^n$ and $G(x) = [g_1(x), \cdots, g_m(x)] \in \mathbf{R}^{n\times m}$ are continuously differentiable. The origin $0 \in \mathbf{R}^n$ is assumed to be the only equilibrium point of (1); that is, $f(0) = 0$. The control objective considered is optimal stabilization. A stabilizing controller $\mu(x)$ drives every trajectory of the closed loop system $\dot{x} = f(x) + G(x)\mu(x)$, $x(0) = x_0$, starting off at $x_0$

[1]This research was partially supported by the AFOSR under contract F49620-95-1-0219 and the NSF under contracts DMI-9625489 and ECS-9612558

[2]The terms **policy** and **controller** are used interchangeably as synonyms throughout the article, both defined as a mapping from the state to the control input. A policy or controller is denoted by the letter $\mu$. The term "policy" is more commonly used in the field of dynamic programming, whereas the term "controller" in the field of control.

at time $t = 0$, asymptotically to the origin, that is, $\lim_{t\to\infty} x(t) = 0$. An optimal controller $\mu^*$ minimizes a *cost function* of the form

$$\int_0^\infty (x(t)^T Qx(t) + u(t)^T Ru(t))dt, \tag{2}$$

for every initial state, over all possible control functions $\mu$, where $Q$ and $R$ are symmetric positive definite matrices. For a given controller $\mu$, the cost function $J_\mu^c : \mathbf{R}^n \to \mathbf{R}$ is defined as $J_\mu^c(x_0) \triangleq \int_0^\infty (x(t)^T Qx(t) + \mu(x(t))^T R\mu(x(t)))dt$, where $x(t)$ denotes the trajectory which starts off at $x(0) = x_0$. A more proper notation should be $x_{x_0}(t)$, but we simplify it. The superscript $^c$ denotes continuous time.

## 2.2 Directional Derivatives

Consider a function $J : \mathbf{R}^n \to \mathbf{R}$, a point $x$ in the state space $\mathbf{R}^n$, and any direction vector $g$ in $\mathbf{R}^n$.

**Definition: Directional Derivative of $J$ along $g$:** The directional derivative $L_g J$ of a function $J : \mathbf{R}^n \to \mathbf{R}$ along the direction of the vector $g$, at a point $x \in \mathbf{R}^n$ is defined as

$$L_g J(x) \triangleq \lim_{\delta\to 0} \frac{J(x + \delta g) - J(x)}{\delta}, \tag{3}$$

provided that the limit exists. Consider a set of vectors, $g_1, \cdots, g_m$, forming a matrix $G = [g_1, \cdots, g_m]$. We denote by $L_G J(x)$ the column vector whose $i$-th element is $L_{g_i} J(x)$, that is,

$$L_G J(x) \triangleq [L_{g_1} J(x), \cdots, L_{g_m} J(x)]^T. \tag{4}$$

In the next section, we use the concept of the directional derivative in order to implement approximate policy iteration.

## 2.3 Discrete time representation of the system dynamics

As discussed above, the approximate policy iteration algorithm requires extensive off-line simulation of the system (1). The discrete time representation that we use for (1) is $x_{t+1} = x_t + \delta(f(x_t) + G(x_t)u_t)$, obtained via a first order Taylor expansion of the function $x(t)$. The closed loop system corresponding to a policy $\mu(x)$ is represented in discrete time by $x_{t+1} = x_t + \delta(f(x_t) + G(x_t)\mu(x_t))$ The integral cost function is replaced by the infinite sum $\sum_{t=0}^\infty \delta(x_t^T Qx_t + u_t^T Ru_t)$, and the integral cost function corresponding to a policy $\mu$ is replaced by the function $J_\mu^d(x_0) \triangleq \sum_{t=0}^\infty \delta(x_t^T Qx_t + \mu(x_t)^T R\mu(x_t))$, where $x_t$ denotes the trajectory which starts off at $x_0$. The superscript $^d$ denotes discrete time. The interval length $\delta$ is fixed throughout the policy iteration process. Since the simulations are done off-line, $\delta$ may be chosen small enough so that the trajectories of the discrete time representation closely resemble the trajectories of (1).

## 2.4 Policy Iteration

As an introduction to the design method, we describe the policy iteration algorithm for the discrete time optimal control problem. This is an *exact* algorithm; there is no approximation. We assume that *a stabilizing controller $\mu_0$ is available such that $J_{\mu_0}^d(x_0)$ is finite for every finite $x_0 \in \mathbf{R}^n$*. The Policy Iteration algorithm generates a sequence of policies $\mu_1, \mu_2, \ldots$ that provably satisfy:

$$J_{\mu_0}^d(x) \geq J_{\mu_1}^d(x) \geq J_{\mu_2}^d(x) \geq \cdots, \quad , \quad \text{for every} \quad x \in \mathbf{R}^n.$$

Starting from the $k$-th policy $\mu_k$, there is a two step process leading to $\mu_{k+1}$:

Policy Evaluation: Compute $J_{\mu_k}^d(x)$ for every $x$;

Policy Improvement: Obtain $\mu_{k+1}(x)$ as:

$$\mu_{k+1}(x) = \arg \min_{u\in\mathbf{R}^m} \{\delta(x^T Qx + u^T Ru) + J_{\mu_k}^d(x + \delta(f(x) + G(x)u))\} \tag{5}$$

At every $x$, determining $\mu_{k+1}(x)$ amounts to a minimization problem over $u$. Clearly, implementation of policy iteration is practically impossible, since both steps of the algorithm involve a computation for every $x \in \mathbf{R}^n$, and the policy improvement step requires the solution of a generally nonconvex optimization problem. This motivates the approximate policy iteration algorithm. Before proceeding, we state two results ensuring that exact policy iteration does asymptotically lead to an optimal controller. This is not a trivial property of the algorithm, and there are classes of optimal decision making problems where it may not be satisfied.

**Existence of an optimal stationary policy:** For this optimal control problem it can be argued on the basis of Corollary 14.1 in [7] that *there exists a stationary optimal policy $\mu^*$*.

**Validity of policy iteration:**

Consider any policy $\mu$ such that $J_\mu^d(x) < \infty$ for every $x$. Then, every trajectory under that policy converges to 0 asymptotically as $t \to \infty$. Therefore, the optimal control problem satisfies the assumptions of Theorem 4.4.1 in [6]: *If $J_{\mu_{n+1}}^d = J_{\mu_n}^d$ for some $n$, then $\mu_n$ is optimal. In general, $\lim_{k\to\infty} J_{\mu_k}^d = J_d^*$, pointwise (where $J_d^*$ is the optimal cost function).*

## 2.5 Approximate Policy Iteration

We consider the case where a stabilizing controller $\mu_0$ is available. In approximate policy iteration, the "policy improvement" step changes name and is called "policy update", since there are no guarantees that the iterate policy is an improvement (or even stable). At the $(k + 1)$-th step of the iteration, given $\mu_k$, we compute the $(k + 1)$-th controller as follows:

Step 1: Approximate Policy Evaluation amounts to determining a function $\tilde{J}_{\mu_k}$ as an approximator of $J_{\mu_k}^d$. We postpone the details of performing this task until

section 2.6. For now, we assume that such an approximation is available and move to the next step.
Step 2: Policy Update amounts to obtaining the updated policy $\mu_{k+1}$ by virtue of (5), where the approximate cost function is in place of $J^d_{\mu_k}$:

$$\mu_{k+1}(x) = \arg\min_{u\in\mathbf{R}^m} \{\delta(x^TQx + u^TRu) + \tilde{J}^d_{\mu_k}(x + \delta(f(x) + G(x)u))\} \quad (6)$$

At every $x$, determining $\mu_{k+1}(x)$ amounts to solving a minimization problem over $u$. However, since $\tilde{J}^d_{\mu_k}$ is in general a nonconvex function, the minimization problem is not easy to solve. To alleviate this problem, notice that differentiability of the architecture allows a Taylor expansion based simplification:

$$\begin{aligned}&\tilde{J}_{\mu_k}(x + \delta(f(x) + G(x)u)) \simeq \\ &\tilde{J}_{\mu_k}(x) + \delta L_{f(x)}\tilde{J}_{\mu_k}(x) + \delta L_{G(x)}\tilde{J}_{\mu_k}(x)\cdot u,\end{aligned} \quad (7)$$

where $L_{G(x)}\tilde{J}_{\mu_k}(x)\cdot u$ is the inner product of a row and column vector of the same size. By replacing (7) in (6), a certain amount of imprecision is introduced, but results in a closed form expression for $\mu_{k+1}$, allowing easy implementation of the updated policy as:

$$\mu_{k+1}(x) = -\frac{1}{2}R^{-1}L_{G(x)}\tilde{J}_{\mu_k}(x) \quad (8)$$

The above form of policy update implementation has been suggested before in [10, 4]. Whereas the imprecision introduced in (7) is guaranteed to be small for small $\delta$, it is questionable whether this is the case after the minimization operation. However, it turns out in Chapter 3 of [3] that the amount of imprecision introduced is of order $o(\delta)$ as $\delta \to 0$.

## 2.6 Approximating the cost function

We suggest two alternative ways for generating an approximation of the cost function $J_{\mu_k}$.

**A. A tuned linear combination of a collection of functions.**

A two stage process is involved in generating $\tilde{J}_{\mu_k}$.
(a) The first stage consists of selecting a sample $x^1, x^2, \ldots$ of the state space, and computing via simulation the value of $J^d_{\mu_k}$ at all those points. To compute $J^d_{\mu_k}(x^i)$, the trajectory starting off at $x^i$ is simulated. (b) In the second stage, we select an architecture to be used for approximation of $J^d_{\mu_k}$. By an approximation architecture we mean a function structure of the form $r_{1,\mu_k}h_1(x) + \cdots + r_{m,\mu_k}h_m(x)$, where $h_i(x)$ are (twice differentiable) nonlinear functions of $x$, and $r_{i,\mu_k}$ are scalar parameters. The architecture is used to interpolate between and extrapolate $J^d_{\mu_k}(x^1), J^d_{\mu_k}(x^2), \cdots, J^d_{\mu_k}(x^N)$ throughout the state space $\mathbf{R}^n$. The parameters $r_{i,\mu_k}$ are selected as the best matching values of the sample $J^d_{\mu_k}(x^1), J^d_{\mu_k}(x^2), \cdots, J^d_{\mu_k}(x^N)$, in a least square sense. Note that the sample covers a bounded region around the origin (semiglobal control). It is selected based on the specification and safety limits of the underlying process. Furthermore, the sample and basis functions are to be *judiciously* selected by the designer. A bad selection is reflected in the approximation error. The choice of the basis functions may be based on engineering insight in relation to a given dynamic system and cost function. A specific case for systems with "partially linear" dynamics is considered in Chapter 5 of [3].

**B. A grid-based approximation architecture**

It follows from the implementation of $\mu_{k+1}$ (8) that it suffices to approximate the directional derivative of $J_{\mu_k}$ in order to generate the $(k+1)$-iterate. The directional derivative $L_GJ_{\mu_k}$ is computed at each vertex of a grid in the state space via numerical integration of a set of differential equations [3]. These values are interpolated throughout the rest of the state space. This approximation strategy comes with the advantage of generality, and without the need for selecting basis functions.

## 3 Conditions for Stability and Cost Improvement after a Single Iteration

### 3.1 Assumptions

We assume that we are given an asymptotically stable closed-loop nonlinear dynamic system of the form

$$\dot{x} = f(x) + G(x)\mu(x), \quad (9)$$

There is a bounded region $X_0$ which includes the origin such that every trajectory of the closed loop system (9) starting off at some point $x_0 \in X_0$ asymptotically converges to the origin, which is the only equilibrium point of (9).
We assume that *All trajectories starting inside $X_0$ belong to a bounded region $X_{inv} \supset X_0$, which is a subset of a compact region $X \supset X_{inv}$.*
The region $X$ represents our best knowledge of $X_{inv}$, since the latter cannot be known exactly. In order to simplify the arguments made in this chapter, we assume that all trajectories of (9) starting inside $X$ asymptotically converge to the origin. We assume that $f(x)$, $G(x)$, $\mu(x)$ are continuously differentiable in $X$, and that $\mu(0) = 0$. We denote by $x_{x_0}(t)$ the trajectory of (9) which starts off at $x_0$ at time 0. That is, $x_{x_0}(0) = x_0$.
**Definition:** The system (9) is **exponentially stable** if there exist a scalar $\beta > 0$ and a scalar $\gamma > 0$ such that, for all $x_0 \in X$, the trajectory $x_{x_0}(t)$ satisfies $\|x_{x_0}(t)\| \leq \beta\|x_0\|e^{-\gamma t}$.
**Assumption:** The system (9) is exponentially stable.
**Lemma:** [3] Consider some $x_0 \in X$. Then, the value of the cost function $J^c_\mu(x_0)$ is finite.
We also make an assumption on the rate of convergence of $x_{x_0}(t)$, namely that it is not faster than exponential.
**Assumption:** There exist positive constants $\beta_1$ and $\gamma_1$ such that $\|x_{x_0}(t)\| \geq \beta_1\|x_0\|e^{-\gamma_1 t}$.

This is not a restrictive assumption, since $\beta_1$ is allowed to be small and $\gamma_1$ is allowed to be large.

**Lemma 3.1** *[3] There exist positive constants $k_1$ and $k_2$ such that*

$$k_1\|x\|^2 \leq x^TQx + \mu(x)^TR\mu(x) \leq k_2\|x\|^2, \tag{10}$$

$$\frac{k_1\beta_1^2}{\gamma_1}\|x\|^2 \leq J_\mu^c(x) \leq \frac{k_2\beta^2}{\gamma}\|x\|^2 \tag{11}$$

*for all $x$ in the set $X$.*

We define the *$a$-level set of $J_\mu^c$ corresponding to a real value* $a \geq 0$ as the locus of points $x$ such that $J_\mu^c(x) \leq a$. It is assumed that there exists a positive $a$ such that the $a$-level set of $J_\mu^c$ is a superset of $X_{inv}$ and a subset of $X$.
Finally:
**Proposition:** [3] If $f$, $G$, $\mu$ are continuously differentiable, then the directional derivative of $J_\mu^c$ in the direction of a vector $g \in \mathbf{R}^n$, $L_gJ_\mu^c(x_0)$, exists and is continuous at any $x_0 \in X$.

## 3.2 Sufficient Condition for Stability of a Single Iterate

We consider the case in which an exponentially stable policy $\mu(x)$ is given, and we perform an approximate policy iteration which results in a new policy $\mu'$. As we saw in Section 1, we approximate $J_\mu^c$ by $\tilde{J}_\mu$ and, recalling definition (4) and (8), the new controller is given by

$$\mu'(x) = [\mu_1'(x), \cdots, \mu_m'(x)]^T = -\frac{1}{2}R^{-1}L_G(x)\tilde{J}_\mu(x) \tag{12}$$

Unless $\tilde{J}_\mu$ is a "good" approximation of $J_\mu^c$, the resulting controller, $\mu'$ is not guaranteed to perform well; it is not even guaranteed to be stable. The result of this section provides a sufficient condition on the approximation error such that stability of $\mu'$ is guaranteed. A thorough discussion of the result is given later, in Section 3.3. Before stating the theorem, note the input weight matrix $R$ can be written in the form $R = R_1^TR_1$, where $R_1$ is some square nonsingular matrix, by virtue of being a symmetric positive definite matrix

**Theorem 3.1** *[3] Assume that $f$, $G$, $\mu$ and $\tilde{J}_\mu$ are continuously differentiable. Let the square nonsingular matrix $R_1$ be such that $R = R_1^TR_1$. If*

$$\begin{aligned}&\frac{1}{2}L_GJ_\mu^c(x)^TR^{-1}[L_GJ_\mu^c(x) - L_G\tilde{J}_\mu(x)] \leq \\ &-\zeta\|x\|^2 + x^TQx + \frac{1}{2}\mu(x)^TR\mu(x) \\ &+\frac{1}{2}[R_1\mu(x) + R_1^{-1}L_GJ_\mu^c(x)]^T[R_1\mu(x) + R_1^{-1}L_GJ_\mu^c(x)],\end{aligned} \tag{13}$$

*for every $x \in X$ and for some positive constant $\zeta$, then the closed loop system*

$$\dot{x} = f(x) + G(x)\mu'(x), \tag{14}$$

*is exponentially stable (in $X_0$) and, therefore, the corresponding cost function $J_{\mu'}^c$ is finite.*

**Elements of Proof:** $J_\mu^c$ is a Lyapunov function for (14).

## 3.3 Discussion and interpretation of the stability criterion

It is notable that the sufficient condition only involves directional derivatives, and not the actual difference between the functions. The importance of approximating the derivative of, rather than the cost function itself, has been emphasized before in [10, 4].

Consider the case where there is no approximation error in the directional derivative. That is, assume that $L_G\tilde{J}_\mu(x) = L_GJ_\mu^c(x)$ for every $x$. In that case, we expect that the policy $\mu'$ is exponentially stabilizing. Indeed, it may be verified that the left hand side of (13) is a sum of negative terms, and that it is smaller than or equal to $-\frac{k_1}{2}\|x\|^2$ according to Lemma 3.1, and thus $J_\mu^c$ is an exponential Lyapunov function for (14). Furthermore, it is obvious that there exists at least a $\frac{k_1}{2}\|x\|^2$ positive margin that can be tolerated. Therefore, condition (13) does allow "not small" approximation errors despite which the updated controller $\mu'$ is stabilizing.

Let us now discuss some aspects of the criterion for the single input case, for simplicity.
**Corollary:** *If the directional derivative $L_g\tilde{J}_\mu(x)$ has the same sign as $L_gJ_\mu^c(x)$, and $|L_g\tilde{J}_\mu(x)| > |L_gJ_\mu^c(x)|$, then*

$$L_gJ_\mu^c(x)\left(L_gJ_\mu^c(x) - L_g\tilde{J}_\mu(x)\right) \leq 0, \tag{15}$$

*which in turn implies that the sufficient condition (13) is satisfied at $x$.*
This shows that if the assumptions of the corollary hold, then (13) is satisfied regardless of the size of $L_g\tilde{J}_\mu(x)$. In other words, there is an infinite gain margin of $[1, \infty)$ in the approximation of $L_gJ_\mu^c(x)$. Let us discuss this from a Lyapunov function point of view. Consider the policy $\mu_e$ generated via policy iteration over $\mu$, assuming zero cost function approximation error, that is, $\mu_e(x) = -\frac{1}{2}R^{-1}L_gJ_\mu^c(x)$. As we saw above, $J_\mu^c$ is a Lyapunov function for the system $\dot{x} = f(x) + g(x)\mu_e(x)$. Let $S(x) = \{y \in \mathbf{R}^n \mid J_\mu^c(y) = J_\mu^c(x)\}$, that is $S(x)$ is a level surface of the Lyapunov function $J_\mu^c$ which goes through $x$. At the point $x$, the vector sum of $f(x)$ and $g(x)\mu_e(x)$ is pointing at the direction where $J_\mu^c$ strictly decreases. According to the corollary, the vector sum of $f(x)$ and $g(x)\mu_m(x)$, for any $\mu_m(x)$ of the same sign and larger absolute value than $\mu_e(x)$, would still point towards the direction of decreasing $J_\mu^c$. It can be easily verified that *exact* policy iteration generates a policy $\mu_e$ which always pushes towards the direction of decreasing $J_\mu^c$. Similar arguments to the above can be made for the multi-input case.

## 3.4 Sufficient Condition for Cost Improvement of a single Iteration

Let a policy $\mu$ such that $J^c_\mu(x) < \infty$ be given, let $\mu'$ given by (12) be the iterate policy. We have:

**Theorem 3.2** *[3] Let $f$, $G$, $\mu$ be continuously differentiable, and let $\tilde{J}_\mu$ be selected twice cont. differentiable, so that $\mu'$ is cont. differentiable. Assume that the error in approximating $J^c_\mu$ by $\tilde{J}_\mu$ is small enough such that the closed loop system under $\mu'$ is exponentially stable, and $J^c_{\mu'}(x) < \infty$ for all $x \in X$. If*

$$\left(\mu'(x)^T R \mu'(x) - \mu(x)^T R \mu(x)\right) + L_G J^c_\mu(x)^T \left(\mu'(x) - \mu(x)\right) \leq 0 \qquad (16)$$

*for $x \in X$, then the performance of the closed loop system under $\mu'$ is improved (in $X_0$) compared to the performance of the closed loop system under $\mu$ with respect to the cost functional (2), that is, $J^c_{\mu'}(x) \leq J^c_\mu(x)$, for all $x \in X$. By plugging in the form $\mu'(x) = -\frac{1}{2}R^{-1}L_G\tilde{J}_\mu(x)$, the sufficient condition (16) takes the form*

$$\left(\frac{1}{4}L_G\tilde{J}_\mu(x)^T R^{-1} L_G \tilde{J}_\mu(x) - \mu(x)^T R \mu(x)\right) + L_G J^c_\mu(x)^T \left(-\frac{1}{2}R^{-1}L_G\tilde{J}_\mu(x) - \mu(x)\right) \leq 0, \qquad (17)$$

*for all $x \in X$.*

## 3.5 Discussion of the cost improvement result

To see whether inequality (17) constitutes a meaningful sufficient condition for improvement, we examine if it holds in the case of no approximation error in the directional derivatives, that is, $L_G\tilde{J}_\mu = L_G J^c_\mu(x)$. Then, it turns out that the left hand side of inequality (17) is equal to

$$-\left[\frac{1}{2}R_1^{-1}L_G J^c_\mu(x) + R_1\mu(x)\right]^T \left[\frac{1}{2}R_1^{-1}L_G J^c_\mu(x) + R_1\mu(x)\right],$$

which is clearly negative. Thus, inequality (17) is automatically satisfied and shows that policy iteration results in a non-deteriorating policy in the case where there is no approximation error in the directional derivative of the cost function. This verifies that Theorem 3.2 gives a sound sufficient condition.

We give an illustrative example of what the criterion on improvement predicts. Consider the case where a system is open loop stable, that is, it is asymptotically stable under the policy $\mu = 0$. For simplicity, assume that the input is scalar and $R = 1$. Assume that at some point $x$, the vector $g(x)$ points towards the direction of decreasing values of $J^c_\mu$, that is, $L_g J^c_\mu(x) < 0$. The improvement criterion (16) for an updated policy $\mu'$ is satisfied if $\mu'(x)^2 + L_g J^c_\mu(x)\mu'(x) = \mu'(x)\left(\mu'(x) + L_g J^c_\mu(x)\right) \leq 0$. Thus, the criterion is satisfied only in the case that $0 \leq \mu'(x) \leq -L_g J^c_\mu(x)$. The improving control is positive, which implies that the vector $g(x)\mu'(x)$ points towards decreasing values of $J^c_\mu$. Therefore, the rate of change of $J^c_\mu$ along trajectories of the closed loop system under the improving controller $\mu'$ is faster than in the open loop case. However, $\mu'$ does not result in improvement if its value exceeds some limit. This is expected, since the size of the control is penalized in the cost function.

## 4 Conclusions

We developed error bounds for approximate policy iteration such that the updated controller after a single iteration is stable, and approximation error bounds such that the continuous time closed loop system under the updated controller after a single iteration is improved with respect to the cost.

## References

[1] D. P. Bertsekas. *Dynamic Programming and Optimal Control,* volume 1. Athena Scientific, Belmont, MA, 1995.

[2] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-dynamic Programming.* Athena Scientific, Belmont, MA, 1996.

[3] C. I. Boussios. *An Approach for Nonlinear Control Design via Approximate Dynamic Programming.* PhD thesis, M.I.T., Cambridge, MA, 1998. Also M.I.T. Lab. for Information and Decision Systems Report No. LIDS-TH-2425.

[4] P. Dayan and S. P. Singh. Improving policies without measuring merits. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8,* pages 1059–1065. MIT Press, Cambridge, MA, 1996.

[5] S. Torkel Glad. Robustness of nonlinear state feedback - a survey. *Automatica,* 23(4):425–435, 1987.

[6] O. Hernandez-Lerma and J. B. Lasserre. *Discrete-Time Markov Control Processes.* Springer-Verlag, New York, 1996.

[7] S. E. Shreve and D. P. Bertsekas. Universally measurable policies in dynamic programming. *Mathematics of Operations Research,* 4(1):15–30, February 1979.

[8] G. Tesauro, D. S. Touretzky, and T. K. Leen, editors. *Advances in Neural Information Processing Systems 7.* MIT Press, Cambridge, MA, 1995.

[9] D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors. *Advances in Neural Information Processing Systems 8.* MIT Press, Cambridge, MA, 1996.

[10] P. Werbos. A menu of designs for reinforcement learning over time. In W. T. Miller IIIrd, R. S. Sutton, and P. Werbos, editors, *Neural Networks for Control,* pages 67–96. MIT Press, Cambridge, MA, 1991.