An Approach for Nonlinear Control Design via Approximate Dynamic Programming

by

Constantinos I. Boussios

Submitted to the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

at the

Massachusetts Institute of Technology

June, 1998

Signature of Author ....
Department of Mechanical Engineering
May 22, 1998

Certified by ....
Munther A. Dahleh
Professor of Electrical Engineering
Thesis Co-Supervisor

Certified by ....
John N. Tsitsiklis
Professor of Electrical Engineering
Thesis Co-Supervisor

Accepted by ....
Anthony T. Patera
Acting Chairman, Departmental Committee on Graduate Students

An Approach for Nonlinear Control Design via Approximate Dynamic Programming

by
Constantinos I. Boussios

Submitted to the Department of Mechanical Engineering
in June, 1998 in partial fulfillment of the requirements for the Degree of
Doctor of Philosophy

Abstract

This thesis proposes and studies a methodology for designing controllers for nonlinear dynamic systems. We are interested in state feedback controllers (policies) that stabilize the state in a given region around an equilibrium point while minimizing a cost functional that captures the performance of the closed loop system. The optimal control problem can be solved in principle using dynamic programming algorithms such as policy iteration. Exact policy iteration is computationally infeasible for systems of even moderate dimension, which leads us to consider methods based on Approximate Policy Iteration. In such methods, we first select an approximation architecture (i.e., a parametrized class of functions) that is used to approximate the cost-to-go function under a given policy, on the basis of cost samples that are obtained through simulation. The resulting approximate cost function is used to derive another, hopefully better policy, and the procedure is repeated iteratively. There are several case studies exploring the use of this methodology, but they are of limited generality, and without much of a theoretical foundation.

This thesis explores the soundness of Approximate Policy Iteration. We address the problem of improving the performance of a given stabilizing controller, as well as the problem of designing stabilizing controllers for unstable systems. For the first problem, we develop bounds on the approximation error that can be tolerated if we wish to guarantee that the resulting controllers are stabilizing and/or offer improved performance. We give bounds on the suboptimality of the resulting controllers, in terms of the assumed approximation errors. We also extend the methodology to the unstable case by introducing an appropriate modification of the optimal control problem.

The computational burden of cost function approximation can be often reduced, thereby enhancing the practicality of the method, by exploiting special structure. We illustrate this for a special class of nonlinear systems with fast linear and slow nonlinear dynamics. We also present an approximation based on state space gridding, whose performance can be evaluated via a systematic test.

Finally, analysis is supported by applying Approximate Policy Iteration to two specific problems, one involving a missile model and the other involving a beam-and-ball model.

Thesis Co-Supervisor: Munther A. Dahleh
Title: Professor of Electrical Engineering

Thesis Co-Supervisor: John N. Tsitsiklis
Title: Professor of Electrical Engineering

Thesis Committee Chairman: Kamal Youcef-Toumi
Title: Professor of Mechanical Engineering

Acknowledgments

I wish to express my deepest thanks and most sincere gratitude to my thesis supervisors, Professors Munther Dahleh and John Tsitsiklis. I wish to thank Munzer for his truly insightful supervision, and for his unique ways of motivating and energizing me. I wish to thank John for his invaluable ideas and feedback and his exceptional guidance throughout this work. Munzer and John's countless and most illuminating suggestions have always broadened my understanding and raised my quality standards and objectives to even higher levels. They have always encouraged me to pursue my own ideas, even those not directly related to the main line of this research. Through their advice, help and encouragement they have brought the best out of me. It is hard to imagine a more rewarding or more enjoyable doctoral thesis experience.

I also wish to thank Professor Kamal Youcef-Toumi, Chairman of my thesis committee. He has been a constant source of guidance, giving me very important suggestions on my research and thesis. I am thankful to Professor Alexandre Megretski for having always been eager to give me his invaluable insight into the research problems I dealt with.

Through the ups and downs of my studies, I have been blessed with the encouragement and support of Professor Elias Gyftopoulos, who has boosted my perseverence and enthusiasm for research. He has been an invaluable mentor to me.

During my years at MIT, I had the good fortune to meet some very remarkable fellow students, postdoctoral associates, and friends. They given me enormous amounts of help by discussing technical issues, sharing their own experiences and advising me, or by listening to my practice presentations. In particular, I am deeply thankful to Christos Athanasiadis, Rienk Bakker, Soosan Beheshti, Harry Doumanidis, Nicola Elia, Ulf Jonsson, Alex Koulouris, Pat Kreidl, Dinos Mavroidis, Wesley McDermott, Kathy Misovec, Reza Olfati-Saber, Fernando Paganini, Leonidas Pantelidis, Babis Papadopoulos, Steve Patek, Sridevi Sarma, Thanos Siapas, Nikos Soukos, George Stamoulis, Mary Tolikas, Ben Van Roy, Manos Varvarigos, Sean Warnick, Ssu-Hsin Yu, Zhenya Zastavker, George Zonios.

I wish to thank my parents from the bottom of my heart for being such great parents. They have always given me their full and unconditional love and support and I have always drawn enormous amounts of courage from them.

Finally, I would like to dedicate this thesis to the two people who inspired, guided and educated me through the course of this work. The fact that I had the chance to work so closely with them has made my whole graduate study experience truly memorable. These people are my thesis supervisors, Professors Munther Dahleh and John Tsitsiklis. I feel extremely privileged and proud to have been their student.

Contents

Chapter 1

Introduction and the Design Method

In this thesis, we are dealing with dynamical systems. As such we describe systems whose state changes in time under some rules - the *dynamics* - that are dictated by the nature of the system. What we refer to as a system in the thesis is a deterministic model (representation) of an underlying physical process. The state of the system is described by a collection of variables that fully characterizes the system. Accordingly, we name this set of variables the *state* of the system, and this is the sole meaning of the term "state" throughout the rest of this document. We consider systems whose state is a vector of real scalars taking values in a Euclidean *state space*, and evolves continuously with time along a continuous *trajectory* through the state space. The value of the state at any time is governed by its past values, as well as by the past values of all external influences to the system. The external influences that are under the control of the system's human or machine supervisor/operator are called *control inputs*. For the class of systems we consider, the control inputs are real scalars collected together in a vector which takes values in a Euclidean *input* *space.* On the other hand, any (typically unpredictable) external forces that might influence the underlying physical process that we model by our system are called *disturbances,* but they are not taken into account in the methodology and analysis presented in this thesis. The task of controlling a dynamical system amounts to selecting the appropriate sequence of control inputs so as to drive the behavior of the dynamic system in a desirable manner. For example, it is imperative to ensure *stability*, which is formally defined as the property that all trajectories of the control system remain bounded. From a practical point of view, the stability requirement amounts to ensuring that the state may not grow beyond some limits imposed either by system performance specification tolerances or even safety considerations. Also, it is desirable to ensure that the state is effectively driven by the control so as to meet system performance specifications. In fact, the control objective is to determine an appropriate control *policy*[1], that is, a function mapping the state space to the input space; the control input at any time is determined via the control policy by the current value of the state. In the physical process level, the state is read with the help of sensors in real time. The controller is in *feedback* form. The resulting controlled system along with the feedback controller form the *closed loop system.* Feedback is desirable because it partly compensates for uncertainty which is present in several forms (e.g. disturbances) in any real world environment. The control *design* task is the task of determining the right feedback policy for our system. As mentioned above, the system that we work with is a model of a physical process, typically obtained via a combination of physical analysis and experiments. The model is assumed to capture the dominant dynamic characteristics of the real life system (device) that we wish to control, although it may not account for every dynamic

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

effect present in real life. The topic of the thesis is a control design methodology for a deterministic dynamic system, typically a model of a more complicated real life process. Although the methodology does not address uncertain systems, it can be shown that the resulting controllers potentially have some desirable performance robustness (with respect to uncertainty) properties which they possess even though such robustness is not among the direct objectives of the deterministic design process. We consider *nonlinear* dynamic systems, whose dynamics are a nonlinear function of the state x and a linear function of the control input u. Nonlinear systems which are linear in the controls include a great variety of systems of interest for which the control design problem is open. Furthermore, it can be shown [67] that an even larger class of systems can be represented by nonlinear systems that are linear in the controls.

The objective of the thesis is to develop a methodology for the design of nonlinear controllers capable of forcing every trajectory of the system to converge towards a desired equilibrium point (the origin $0 \in \mathbf{R}^n$). This problem is called *stabilization*. In addition, the method aims at designing controllers which are optimal with respect to some trajectory performance criterion. The performance criterion weighs the distance of the state from the targeted equilibrium point at any time during a trajectory, as well as the amount of control input applied at any time during the trajectory.

The stabilization objective captures more general command following problems, which can be brought into the stabilization formulation by appropriate state variable transformations. The combined stabilization/optimal control objective also captures the stability problem, since it specifies that the state is forced close to a desired point of operation.

As far as the problem of stabilization goes, there exist several approaches to the nonlinear control design problem. Many apply to special classes of systems. We mention feedback linearization [65, 37, 76], which applies to nonlinear systems that can be transformed to linear ones under an appropriate feedback law, and the transformed linear system can be addressed by one of many available linear control techniques. The backstepping algorithm [54, 55] is a control design technique applied to a class of feedback linearizable systems with special structure. For systems whose dynamics include a component with linear corresponding dynamics, the gain scheduling approach [78, 57, 58, 53] produces controllers that are linear in the part of the state with linear dynamics and nonlinear in the rest. It is most effective in cases where the linear part of the dynamics is dominant. More advanced recent variants [6, 50, 47, 46] of gain scheduling have led to effective and computationally efficient controller design tools by taking advantage of the special structure. Other control design approaches apply to more general classes of nonlinear systems. Sliding mode control forces the state trajectory to approach an attractive invariant manifold, the *sliding surface*, which goes through the origin. Once the trajectory reaches the sliding surface, it is forced to move along it towards the origin. Sliding mode control [62, 74] has found many applications [31, 64], but it comes with potential problems like large excursions away from the origin before reaching the sliding surface, high control input requirements, and excitation of high frequency unmodeled dynamics during the slide (the latter can be compensated for by forcing the state to a "boundary layer" around the sliding surface rather than the surface itself [61], at the expense of tracking error).

The above methodologies and others which were not mentioned typically fail to address the issue of performance. On the contrary, some of them produce controllers that are by design of potentially poor performance. For example, gain scheduled controllers force the trajectories to move close to a certain surface of the state space around which linearity is practically valid, and at the same time slowly so as to diminish the effect of nonlinearity. There may exist preferable alternative trajectory routes, but gain scheduling control does not even seek to follow them. As far as feedback linearizing control goes, its primary objective, that is linearization, is not related to any acceptable trajectory performance measure, whereas it can be easily demonstrated that performance may potentially be seriously degraded.

The optimal control approach, on the other hand, does address the issue of performance. Depending on the exact form of the criterion with respect to which a controller is optimal, it potentially produces trajectories with desirable features, like fast convergence to the origin, absence of oscillation, low control input requirements, etc. On a different twist, it has been shown that optimal nonlinear controllers possess some desirable robustness properties with respect to errors in modeling of the underlying physical process [73, 26, 27]. It can be shown that for a large range of modeling errors (nonlinear *gain* and *phase margins*), an optimal controller designed for the inaccurate model of a physical process is guaranteed to be stabilizing if applied to the undermodeled physical process. This is a quite attractive feature from a practical point of view, and constitutes one of the main motivations behind working with optimal control based design methods.

Finding analytical solutions to the nonlinear control design problem is a nontrivial task. It has only been achieved satisfactorily for limited classes of systems. Even more difficult a task is to analytically solve the optimal control problem. On the positive side, the optimal controller can be characterized by means of a partial differential equation in the state variables, the Hamilton-Jacobi (HJ) equation. The solution of the HJ equation is a control Lyapunov function of the dynamic system, on the basis of which one can design a stable control law. The HJ equation is analytically intractable, except in very few cases. Several optimal control approaches focus on approximately solving the HJ equation computationally [21, 42, 29]. Unfortunately, any such approach suffers from the *curse of dimensionality*, that is, the tremendous amount of computation required to solve the HJ equation exactly. These approaches work well for problems of low dimension, like 2 or 3, whereas the computational complexity increases exponentially when one moves to higher dimensions.

Another computational approach to solving the optimal control problem is based on dynamic programming algorithms. Dynamic programming is a collection of algorithms which address the optimal control problem for dynamic systems in *discrete time* [8, 36, 9, 10]. The state of a discrete time system changes at discrete instances of time (a system whose state varies continuously with time can be approximated by an appropriate discrete time system, as discussed in detail later). Again, dynamic programming algorithms suffer from the curse of dimensionality. Therefore, approximate versions of these algorithms are more appropriate.

Several approaches for nonlinear control using **Approximate Dynamic Programming** algorithms have been proposed. Recent surveys of these algorithms can be found in [80, 11, 68]. The focus is on developing practical algorithms. Unfortunately, these algorithms come with no study on the effect of approximation on stability and performance of the resulting controllers. Some theoretical results include [11], that provides bounds on suboptimality of the generated policies for cases of decision problems where instability is not a possibility and the cost is discounted, and [17] for a linear quadratic problem in which case the exact form of the function to be approximated is known, and all that is needed is tuning some parameters to their right values. In most cases, justification of the method's validity is based on some successful control design for a specific application. Overviews of recent research are given in [80, 71, 72]. However, the lack of theoretical results undermines the credibility of the method as a general design tool for "difficult" problems which cannot be effectively addressed by other design methods.

In this thesis, we propose *Approximate Policy Iteration*, one of the Approximate Dynamic Programming algorithms as a nonlinear control design tool. Approximate policy iteration is a computational method based on off-line simulation of the dynamic system. It has been successfully used in several applications [80, 71, 72], mainly for problems with non-Euclidean state spaces. It assumes that a stabilizing controller is available (actually, a controller for which the cost accumulated along any trajectory of the corresponding closed loop system is finite). Given the controller, the closed loop system is discretized and a number of trajectories is simulated off-line. The cost of each trajectory is computed. The *cost function* is a function mapping a value of the state x to the cost corresponding to the trajectory that starts at x. The simulations provide a finite number of samples of the cost function. The approximation task amounts to somehow determining an approximate cost function, based on the finitely many samples of the cost function via interpolation and extrapolation. An updated controller is then obtained based on the approximate cost function, as described later. The same process is iterated until a satisfactory controller is finally obtained. As will be discussed in Chapter 4, in the case that no stabilizing controller is known for a system, then a proper modification of the algorithm just described is possible which uses approximate policy iteration to obtain a stabilizing controller.

The thesis objective is to establish the method's credibility and control engineers' confidence in it via theory and computational experiments. In the next section, we define the problem. In Section 1.2 we present the proposed algorithm; Finally, in Section 1.3 we give an overview of the thesis and discuss our objectives and contributions.

1.1 Problem Formulation

We consider a continuous time nonlinear dynamic system of the form

$$\dot{x} = f(x) + G(x)u, \tag{1.1}$$

where $\dot{x}$ denotes the derivative of the state with respect to time, dx/dt; $f : \mathbf{R}^n \to \mathbf{R}^n$ denotes the state dynamics, a nonlinear continuously differentiable function of the state x; The matrix

$$G(x) = [g_1(x), \cdots, g_m(x)] \in \mathbf{R}^{n \times m}$$

is called the *input matrix*, consisting of the continuously differentiable nonlinear column vector functions $g_i : \mathbf{R}^n \to \mathbf{R}^n$, $i = 1, \ldots, m$. The origin $0 \in \mathbf{R}^n$ is assumed to be the only equilibrium point of (1.1); 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

$$\dot{x} = f(x) + G(x)\mu(x), \qquad x(0) = x_0, \tag{1.2}$$

starting off at x_0 at time $t = 0$, asymptotically to the origin, that is, $\lim_{t\to\infty} x(t) = 0$. An optimal controller μ^* minimizes a *cost function* of the form

$$\int_0^\infty (x(t)^T Q x(t) + u(t)^T R u(t))dt, \tag{1.3}$$

for every initial state, over all possible control functions μ, where Q and R are symmetric positive definite matrices. For a given controller μ, the cost function $J^c_\mu : \mathbf{R}^n \to \mathbf{R}$ is defined as

$$J^c_\mu(x_0) = \int_0^\infty (x(t)^T Q x(t) + \mu(x(t))^T R \mu(x(t)))dt, \tag{1.4}$$

where $x(t)$ denotes the trajectory which starts off at x_0 at time 0, that is, $x(t = 0) = x_0$. A more proper notation should be $x_{x_0}(t)$, but we simplify it. The superscript ^{c} denotes continuous time.

1.2 Approximate Policy Iteration

1.2.1 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 1.2.1 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*

$$L_g J(x) \triangleq \lim_{\delta \to 0} \frac{J(x + \delta g) - J(x)}{\delta}, \tag{1.5}$$

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,

$$L_G J(x) \triangleq [L_{g_1} J(x), \cdots, L_{g_m} J(x)]^T. \tag{1.6}$$

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

1.2.2 Discrete time representation of the system dynamics

As discussed above, the approximate policy iteration algorithm requires extensive off-line simulation of (1.1). Therefore, we discuss how the system is represented in discrete time, before presenting the algorithm. The discrete time representation that we use for (1.1) is

$$x_{t+1} = x_t + \delta(f(x_t) + G(x_t)u_t). \tag{1.7}$$

This is 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

$$x_{t+1} = x_t + \delta(f(x_t) + G(x_t)\mu(x_t)) \tag{1.8}$$

The integral cost function (1.3) is replaced by the infinite sum

$$\sum_{t=0}^{\infty} \delta(x_t^T Q x_t + u_t^T R u_t), \tag{1.9}$$

and the cost function (1.4) corresponding to a policy μ is replaced by the function J_μ^d given by

$$J_\mu^d(x_0) = \sum_{t=0}^{\infty} \delta(x_t^T Q x_t + \mu(x_t)^T R \mu(x_t)). \tag{1.10}$$

where x_t denotes the trajectory which starts off at x_0 at time 0, that is, $x_{t=0} = x_0$. A more proper notation should be $x_t^{x_0}$, but we simplify it. The superscript ^{d} denotes discrete time. We keep the interval length δ fixed throughout the policy iteration process. In practice, it may be desirable to vary it at different stages of the policy iteration process, but by assuming that it is kept constant we simplify presentation of the results.

In general, first order Taylor expansion for the dynamics is not a sound discretization strategy, unless δ is sufficiently small and the functions f, G, and μ are sufficiently smooth. Here, a brute force approach is assumed. Since the simulations are done off-line, we choose δ as small as needed so that the trajectories of (1.8) closely resemble the trajectories of (1.1). To determine whether a certain value of δ is small enough, we compare trajectories of (1.8) to Runge-Kutta simulations of (1.1). Of course, if we are forced to choose a very low value of δ, we run into problems of excessive computational requirements. This issue, however, is beyond the scope of the thesis. Our experience with applications of the algorithm (Chapters 7 and 8) indicates that this is not a serious issue for many interesting practical cases. Thus, we have

Assumption 1.2.1 *The discretization interval δ is judiciously chosen small enough so that the representation (1.8) and (1.9) captures all the important attributes of (1.1) and (1.3)*

In the next sections we describe the design method. We only consider the case where we are given an initial stabilizing controller μ_0, such that $J^c_{\mu_0}(x_0)$ (and $J^d_{\mu_0}(x_0)$, based on Assumption 1.2.1) is finite for every finite $x_0 \in \mathbf{R}^n$, and we seek to design improved controllers, with respect to the specified cost. The case in which no stabilizing controller is available and we have to work with an open loop unstable system is considered separately in Chapter 4.

1.2.3 Policy Iteration

As an introduction to the design method, we describe the policy iteration algorithm for the discrete time optimal control problem (1.8), (1.9). This is an *exact* algorithm; there is no approximation. Approximations are introduced in the approximate policy iteration algorithm, which we present in the next section.

Assumption 1.2.2 *A stabilizing controller μ_0 is available such that $J^d_{\mu_0}(x_0)$ is finite for every finite $x_0 \in \mathbf{R}^n$.*

The Policy Iteration algorithm generates a sequence of policies $\mu_1, \mu_2, \cdots$ that provably satisfy:

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

Starting from the k-th policy μ_k, there is a two step process leading to μ_{k+1}:
Policy Evaluation: Compute $J^d_{\mu_k}(x)$ for every x;

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

$$\mu_{k+1}(x) = \arg \min_{u \in \mathbf{R}^m} \left\{ \delta(x^T Q x + u^T R u) + J^d_{\mu_k} (x + \delta(f(x) + G(x)u)) \right\} \qquad (1.11)$$

At every x, determining $\mu_{k+1}(x)$ amounts to a minimization problem over u. Let's take a closer look at the Policy Improvement step. Assume that at time $t = 0$ the value of the state is x, and that we have the option to freely choose a control input u_0 at $t = 0$, but for all $t \geq 1$ we are forced to use $u_t = \mu_k(x_t)$. Then, the best way to take advantage of the option to freely select u_0 is

$$u_0 = \arg \min_{u \in \mathbf{R}^m} \left\{ \delta(x^T Q x + \mu(x)^T R \mu(x)) + J^d_{\mu_k} (x + \delta(f(x) + G(x)u) \right\}.$$

The trajectory with u_0 as above at $t = 0$ and $u_t = \mu_k(x_t)$ thereafter, is of improved cost in comparison to the one resulting from $u_t = \mu_k(x_t)$ for all $t \geq 0$. Applying the same strategy to select an control at every x, results in the improved controller μ_{k+1}.

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. However, before we move on we cite two results which ensure that it makes sense to use a policy iteration algorithm for optimal control. They are technical results ensuring that if we do exact policy iteration, we will asymptotically find an optimal controller. Notice that 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:

The optimal control problem (1.7)- (1.9) has the following features:

1. Consider the space $\Gamma = \mathbf{R}^n \times \mathbf{R}^m$, that is, the cross product between the state and control spaces. It turns out that Γ can be written in the form

$$\Gamma = \cup_{j=1}^{\infty} \Gamma^j,$$

for a sequence of compact sets $\Gamma^1 \subset \Gamma^2 \subset \ldots$.

2. The stage cost function $x^T Q x + u^T R u$ is continuous on Γ.

3. The stage cost function satisfies

$$\lim_{j \to \infty} \inf_{(x,u) \in \Gamma^j - \Gamma^{j-1}} \left\{ x^T Q x + u^T R u \right\} = \infty.$$

4. The dynamics $x + \delta(f(x) + G(x)u)$ are continuous in x, u.

For this optimal control problem it can be argued on the basis of Corollary 14.1 in [60] that **there exists a stationary optimal policy** μ^*.

Validity of policy iteration:

Consider any policy μ 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 [35], which was originally shown in [34]. According to it, we have:

Lemma 1.2.1 (Based on Hernardez-Lerma and Munoz de Ozak, 1992)

1. *If $J^d_{\mu_{n+1}} = J^d_{\mu_n}$ for some n, then μ_n is optimal.*

2. *In general, $\lim_{k \to \infty} J^d_{\mu_k} = J^*_d$, pointwise (where J^*_d is the optimal cost function).*

1.2.4 Approximate Policy Iteration

We consider the case where a stabilizing controller μ_0 is available and Assumption 1.2.2 holds. Approximate policy iteration is again a two-step algorithm. However, the second step now changes name and is called the "policy update" step, since there are no guarantees that the new policy is an improvement. Quite the contrary, for large approximation errors it might even turn out to be unstable. Assume that we are at the $(k+1)$-th step of the iteration, the stabilizing controller μ_k is available and we want to compute the $(k+1)$-th controller:

Step 1: Approximate Policy Evaluation amounts to determining a function $\tilde{J}_{\mu_k}$ as an approximator of $J^d_{\mu_k}$. This is a two stage process.

(a) The first stage consists of selecting a sample $x^1, x^2, \ldots$ of the state space, and computing via simulation (with the use of (1.8) and (1.9)) the value of $J^d_{\mu_k}$ at all those points, $J^d_{\mu_k}(x^1), J^d_{\mu_k}(x^2), \cdots, J^d_{\mu_k}(x^N)$. To compute $J^d_{\mu_k}(x^i)$, we simulate the trajectory of (1.8) which starts off at x^i. We simulate it for a long enough time, T_i, such that the sum $\sum_{t=0}^{T_i} \delta(x_t^T Q x_t + \mu_k(x_t)^T R \mu_k(x_t))dt$ has essentially converged to the value $J^d_{\mu_k}(x^i)$.

(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

$$r_{1,\mu_k} h_1(x) + \cdots + r_{m,\mu_k} h_m(x), \tag{1.12}$$

where $h_i(x)$ are nonlinear functions of x, and r_{i,μ_k} are scalar parameters, whose values are to be selected such that (1.12) best matches the actual cost $J^d_{\mu_k}$. The architecture, with the best values of the coefficients r_{i,μ_k}'s 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 $\mathbf{R}^n$. We choose this architecture to be twice differentiable, that is, every h_i is twice differentiable. This allows us to perform the policy evaluation step in a convenient way, as shown in the sequel. It also leads to a differentiable μ_{k+1}. Given the collection of samples

$J^d_{\mu_k}(x^1), J^d_{\mu_k}(x^2), \cdots, J^d_{\mu_k}(x^N)$, and having selected an architecture (1.12), we select the parameters r_{i,μ_k} such that (1.12) best matches $J^d_{\mu_k}(x^1), J^d_{\mu_k}(x^2), \cdots, J^d_{\mu_k}(x^N)$ in a least square sense:

$$r_{\mu_k} = \arg \min_{r \in \mathbf{R}^m} \| \, [J^d_{\mu_k}(x^1), \ldots, J^d_{\mu_k}(x^N)] - [\tilde{J}_{\mu_k}(x^1, r_{\mu_k}), \ldots, \tilde{J}_{\mu_k}(x^N, r_{\mu_k})] \, \|_2 \quad (1.13)$$

The last minimization is a standard linear least squares problem, and can be easily computed.

Remark 1: Notice that according to Assumption 1.2.1, $\tilde{J}_{\mu_k}$ is about as good an approximation of $J^c_{\mu_k}$ as of $J^d_{\mu_k}$, and it is viewed as such in the thesis.

Remark 2: The sample $x^1, \cdots, x^N$ clearly cannot cover but a bounded region of the state space, including the origin. Outside that region, the cost function is extrapolated. Obviously, the quality of approximation outside that bounded region is potentially very poor. This is a limitation of every computational method. However, this does not pose any serious limitation, from a practical point of view, since in practice the system may only operate in a bounded region around the origin. This region is defined by specification limits, safety limits, and/or input saturation limits. This is the region of interest, for which sound cost function approximation is crucial.

Remark 3: We do not propose any systematic way of appropriately selecting the samples $x^1, \cdots, x^N$. This is beyond the scope of the thesis. The samples are left to the *judicious* selection of the control designer, who has to make sure that it is evenly distributed so as to adequately represent the whole region of interest. The effectiveness of the sample choice influences the success of the approximation. A bad selection is reflected in the approximation error. In the thesis we derive several conditions on the approximation error such that approximate policy iteration produces good controller designs. The sample selection is indirectly accounted for via those conditions on the approximation error.

Remark 4: The choice of the "basis functions" h_i is also based on engineering insight in relation to a given dynamic system and cost function. In Chapter 6, we propose an alternative, systematic way for generating an approximation, which comes with the advantage of generality and the potential disadvantage of increased computational complexity since it uses no insight that might simplify the cost function approximation task. Our decision to present our material with architectures of the type (1.12) in the first few chapters is due to its relative simplicity in comparison to the systematic architecture that is introduced in Chapter 6.

Step 2: Policy Update amounts to obtaining the updated policy μ_{k+1} by virtue of (1.11), where the approximate cost function is in place of the actual cost function $J^d_{\mu_k}$:

$$\mu_{k+1}(x) = \arg \min_{u \in \mathbf{R}^m} \left\{ \delta(x^T Q x + u^T R u) + \tilde{J}^d_{\mu_k}\left(x + \delta(f(x) + G(x)u)\right) \right\} \tag{1.14}$$

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 in general. To alleviate this problem, notice that differentiability of the architecture allows us to write the following Taylor expansion based simplification:

$$\tilde{J}_{\mu_k}(x + \delta(f(x) + G(x)u)) \simeq \tilde{J}_{\mu_k}(x) + \delta L_{f(x)} \tilde{J}_{\mu_k}(x) + \delta L_{G(x)} \tilde{J}_{\mu_k}(x) \cdot u, \tag{1.15}$$

where the vector of directional derivatives $L_G J$ of a function J along directions $g_1, \cdots, g_m$ has been defined in (1.5) and (1.6), and 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. With the above simplification, we implement the policy update step as

$$\mu_{k+1}(x) \quad = \arg\min_u \{ \quad \delta(x^T Q x + u^T R u) + \tilde{J}_{\mu_k}(x) + \delta L_{f(x)} \tilde{J}_{\mu_k}(x)$$

$$+\delta \cdot L_{G(x)}\tilde{J}_{\mu_k}(x) \cdot u\}$$

$$= \arg\min_u\{ \quad u^T R u + L_{G(x)}\tilde{J}_{\mu_k}(x) \cdot u\} \tag{1.16}$$

Although a certain amount of imprecision was introduced in (1.16), as compared to (1.14), it is beneficial because it results in a convex minimization problem. The latter actually has a closed form solution, which allows easy implementation of the new policy μ_{k+1} as follows:

$$\mu_{k+1}(x) = -\frac{1}{2}R^{-1}L_{G(x)}\tilde{J}_{\mu_k}(x) \tag{1.17}$$

The above form of policy update implementation has been suggested before by [79, 23]. Whereas the imprecision introduced in (1.15) is guaranteed to be small for small δ, it is questionable whether this is the case after the minimization operation. However, it turns out in Chapter 3 that the amount of imprecision introduced by replacing (1.14) by (1.16) is of order $o(\delta)$ as $\delta \to 0$, as revealed by relation (3.11) and the arguments following (3.11).

Finally, the updated closed loop system is given by

$$\dot{x} = f(x) - G(x) \cdot (\frac{1}{2}R^{-1}L_{G(x)}\tilde{J}_{\mu_k}(x)) \tag{1.18}$$

1.3 Thesis Outline

The thesis objective amounts to addressing two basic questions:
(a) **Feasibility** of approximate policy iteration as a practically useful design method, which can be used in a real world of limited computational capacity and, consequently, of limited function approximation capabilities.
(b) Potential **viability** of the method as a control design tool; even with the feasibility

question answered, practicality of the method for control design depends on whether the task of a multivariable function approximation can be carried out relatively easily in an application. We propose ways in which the structure of the dynamics can be used to relax the approximation burden; we also suggest a general architecture, which can be used systematically with the advantage of a systematic way to evaluate the quality of the approximation; and, finally, we report our application experience with the method, which is positive and quite convincing.

In detail, **Chapters 2** and **3** mainly address the question of feasibility. In Chapter 2 we develop bounds on the allowed approximation error $\tilde{J}_{\mu_k} - J^c_{\mu_k}$ under which the resulting closed loop system controlled by μ_{k+1} is exponentially stable. We also develop bounds on the allowed approximation error under which the resulting closed loop system controlled by μ_{k+1} has better performance than the closed loop system under μ_k with respect to the cost of the form (1.3).

In Chapter 3 we develop an upper bound on the suboptimality of the controllers resulting from the method as a function of the approximation errors at each iteration of the algorithm. The bound is conservative, and therefore not of great practical importance, but certainly of conceptual importance since it establishes the overall soundness of the approach.

The second question is addressed primarily in Chapters 5 and 6: In **Chapter 5**, we consider a class of nonlinear systems with a special structure. Based on the structure, we propose a special approximation architecture which makes the cost function approximation less complex. We show an analytical result that provides a proof of concept.

In **Chapter 4**, we address the case where the open loop system is unstable. We modify the problem (1.8)- (1.9) appropriately so that we always have an initial policy with finite cost, even though no stabilizing controller is a priori available for the system. We show theoretically the feasibility of the approach, and we discuss its implementation.

In **Chapter 6**, we propose a particular architecture, which can be used systematically for approximation. For this architecture, in conjuction with the stability criterion of Chapter 2, we develop a systematic criterion for evaluating the function approximation.

In **Chapter 7**, we apply the policy iteration method to a missile control problem. We evaluate the general method, as well as the simplification ideas of Chapter 5. Finally, in **Chapter 8** we report the application of the method to the beam-and-ball problem. This is an unstable system, and it is widely used in the nonlinear control academic community as a benchmark for evaluation. For approximation, the architecture proposed in Chapter 6 is used.

Chapter 2

Conditions for Stability and Cost Improvement after a Single Iteration

Whereas a single *exact* policy iteration is guaranteed to produce a controller of improved performance with respect to the cost functional (1.9), no guarantees are provided by a single *approximate* policy iteration. Poor cost function approximation may result in a poorly performing controller. Even worse, it may result in an unstable controller. Clearly, the soundness of approximate policy iteration relies on the designer's approximation capabilities. Past attempts to use the algorithm or its variants for optimal control in certain applications have naturally recognized that fact. However, there has never been an effort to come up with a general soundness criterion, that is, a sufficient condition on the quality of approximation such that the controller obtained by the iteration is stable. In this chapter, we develop such conditions. The conditions provide some valuable insight into approximate policy iteration, which will be discussed in the sequel. Furthermore, in Chapter 6, we will show that the conditions can be transformed into a systematic tool for checking the soundness of approximate policy iteration for any given application.

2.1 Description and Assumptions on the System

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

$$\dot{x} = f(x) + G(x)\mu(x), \tag{2.1}$$

where the dimensions and form of x, $f(x)$, $G(x)$, $\mu(x)$ are described in Section 1.1. There is a bounded region X_0 which includes the origin such that every trajectory of the closed loop system (2.1) starting off at some point $x_0 \in X_0$ asymptotically converges to the origin, which is the only equilibrium point of (2.1). As discussed in Chapter 1, the region X_0 represents the area of operation of the underlying physical system, which is always bounded. From the point of view of approximate policy iteration, the requirement of simulating system trajectories imposes the restriction of designing controllers for a bounded region around the origin, which can be selected as large as desired. We state the following assumption:

Assumption 2.1.1 *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 (2.1) starting inside X asymptotically converge to the origin. Now, we make assumptions on the differentiability properties of the dynamics.

Assumption 2.1.2 *We assume that $f(x)$, $G(x)$, $\mu(x)$ are continuously differentiable in X.*

Assumption 2.1.2 is needed in order to make the criterion that we develop simple. It may look restrictive from a practical point of view, but it turns out that it is not as restrictive as it seems. In practice, the dynamics of many systems may not be differentiable. As examples we mention systems with saturations in the control inputs, and gain-scheduled controllers. These systems exhibit non-differentiability on some $(n-1)$-dimensional surfaces, away from the origin. However, such non-differentiable dynamics can often be approximated (and thus modeled) arbitrarily well by a differentiable or even smooth dynamic model. A smooth model is actually a better representation of the real world in many cases. For example, actuator saturations are typically smooth nonlinearities modeled by a nondifferentiable saturation function for model simplicity. Furthermore, in the case that a controller is nondifferentiable away from the origin, it may be replaced in practice by a differentiable one. Similarly, there are cases where a controller is discontinuous on some $(n-1)$-dimensional surfaces, away from the origin. Again, such a controller may in many cases be replaced by a differentiable controller [46]. In a different scenario, it is a proven fact that some nonlinear systems admit discontinuous stabilizing controllers, whereas they may not be stabilized by any continuous controller [18, 5, 75]. In many of these cases, any stabilizing controller is discontinuous on a $(n-1)$-dimensional surface passing through the origin, such that the dynamics $f(x)+G(x)\mu(x)$ near the surface, point away from it. Consequently, the surface essentially divides the state space into two separate parts, and our analysis would apply to each of those two parts.

In the same spirit, there are systems that do not admit continuously differentiable controllers [13, 14]. For example, there are systems for which any stabilizing controller has to be nondifferentiable at the origin [63]. We believe that our analysis and results for the differentiable case generalize to many such cases. For some, we might have to use a different form of cost functional (for example, higher even powers of x and/or u). However, we do not pursue those directions in the thesis, partly because the continuously differentiable case encompasses a very large number of interesting nonlinear control problems, including those for which we argued in the previous paragraph, partly because the purpose of the thesis is developing some basic ideas rather than exhaustively addressing all the different classes of nonlinear systems which can be possibly addressed by the same ideas, and, finally, partly because the main challenge of such extensions is purely mathematical and might not offer much insight into the control problem.

Next, we give the definition of exponential stability, which in conjuction with continuous differentiability of μ implies that the cost function $J_\mu^c(x_0)$ is finite for finite x_0. The exponential stability assumption proves very useful in this chapter, by helping us give simple and clear technical mathematical arguments.

Definition 2.1.1 of Exponential Stability *Let us denote by $x_{x_0}(t)$ the trajectory of (2.1) which starts off at x_0 at time 0. That is, $x_{x_0}(0) = x_0$, for any $x_0 \in \mathbf{R}^n$. The system (2.1) 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*

$$||x_{x_0}(t)|| \leq \beta ||x_0|| e^{-\gamma t}. \tag{2.2}$$

Assumption 2.1.3 *The system (2.1) is exponentially stable.*

Lemma 2.1.1 *Let the system (2.1) satisfy Assumptions 2.1.2 and 2.1.3. Consider some $x_0 \in X$. Then, the value of the cost function $J_\mu^c(x_0)$ is finite:*

$$J_\mu^c(x_0) = \int_0^\infty (x_{x_0}(t)^T Q x_{x_0}(t) + \mu(x_{x_0}(t))^T R \mu(x_{x_0}(t)))dt < \infty, \tag{2.3}$$

Proof:

Consider $x_0 \in X$. From exponential stability it follows directly that

$$\int_0^\infty \left(x_{x_0}(t)^T Q x_{x_0}(t)\right) dt < \infty. \tag{2.4}$$

From exponential stability, it follows that the trajectory $x_{x_0}(t)$ is bounded. Therefore, from the continuous differentiability of μ and the fact that $\mu(0) = 0$, it follows that there exists a constant $M_{x_0} > 0$ such that

$$||\mu(x_{x_0}(t))|| \leq M_{x_0} ||x_{x_0}(t)||, \qquad \forall \quad t.$$

Therefore,

$$\int_0^\infty \mu(x_{x_0}(t))^T R \mu(x_{x_0}(t)))dt < \infty. \tag{2.5}$$

From (2.4) and (2.5), it follows that

$$J_\mu^c(x_0) < \infty.$$

■

Remark: There are cases where the system (2.1) may not be exponentially stable, but $J_\mu^c(x_0)$ is still finite. If $J_\mu^c(x_0)$ is infinite, we may use a different cost function for our optimal control problem, involving higher even powers of x and/or $\mu(x)$, and $J_\mu^c(x_0)$ may then become finite. Results similar to those shown in this thesis would also be possible with such a cost function. Selecting a quadratic cost function and assuming exponential stability in this thesis is adopted only in order to simplify presentation of the ideas and concepts suggested.

We also make the following assumption on the rate of convergence of $x_{x_0}(t)$, namely that it is not faster than exponential.

Assumption 2.1.4 *There exist positive constants β_1 and γ_1 such that*

$$\|x_{x_0}(t)\| \geq \beta_1 \|x_0\| e^{-\gamma_1 t}. \tag{2.6}$$

This is not a restrictive assumption, since β_1 is allowed to be small and γ_1 is allowed to be large.

Next, we state a lemma on the shape of the integrand in the cost function, $x^T Qx + \mu(x)^T R\mu(x)$. We remind the assumption that $\mu(0) = 0$, which is justified by the fact that the equilibrium point 0 is the targeted point of operation, and once there, the state is desired to remain there.

Lemma 2.1.2 *There exist positive constants k_1 and k_2 such that*

$$k_1\|x\|^2 \leq x^T Qx + \mu(x)^T R\mu(x) \leq k_2\|x\|^2, \quad , \tag{2.7}$$

for all x in the set X.

This result is used in the exponential stability arguments involved in the proof of Theorem 2.3.1. A detailed proof of Lemma 2.1.2 is omitted. The lower bound proof is straightforward. The upper bound proof draws from differentiability of $\mu(x)$ at the origin, which implies that $\mu(x)$ looks roughly linear close to 0, and therefore the ratio $\frac{\mu(x)^T R\mu(x)}{\|x\|^2}$ is bounded. Clearly, this is the case away from the origin as well, and the argument follows.

From Assumptions 2.1.3, 2.1.4 and Lemma 2.1.2 it follows that

$$k_1\beta_1^2\|x_0\|^2 e^{-2\gamma_1 t} \leq \mu(x_{x_0}(t))^T R\mu(x_{x_0}(t)) + x_{x_0}(t)^T Q x_{x_0}(t) \leq k_2\beta^2\|x_0\|^2 e^{-2\gamma t} \tag{2.8}$$

From the last pair of inequalities we have the following corollary on the shape of the cost function J_μ^c in X_{inv}:

Corollary 2.1.1 *The cost function J_μ^c satisfies*

$$\frac{k_1\beta_1^2}{\gamma_1}\|x_0\|^2 \leq J_\mu^c(x_0) \leq \frac{k_2\beta^2}{\gamma}\|x_0\|^2 \tag{2.9}$$

for all $x_0 \in X$.

Before stating the last assumption, let us give the definition of a *a-level set of J_μ^c corresponding to a real value $a \geq 0$* as the locus of points x such that $J_\mu^c(x) \leq a$.

Assumption 2.1.5 *There exists a positive a such that the a-level set of J_μ^c is a superset of X_{inv} and a subset of X.*

The significance of this assumption is discussed in the proof of Theorem 2.3.1.

2.2 Directional Derivatives of the Cost Function

In this section, we prove that the directional derivative of the cost function J_μ^c along a direction g exists and is continuous at all $x \in X$. Note that the result is proved for any fixed vector g, and there should be no confusion with the input vector notation $g(x)$. For notational simplification, we define the *closed loop dynamics (2.1) under policy μ, F_μ,* as

$$F_\mu(x) \triangleq f(x) + G(x)\mu(x). \tag{2.10}$$

This notation is used throughout this section. The function F_μ is continuously differentiable, and $F_\mu(0) = 0$.

Consider a point $x_0 \in X$. Let us denote by $x(t, \delta)$ the trajectory of (2.1) which starts off at $x_0 + \delta g$ at time $t = 0$, for every δ such that $|\delta| \leq 1$:

$$\frac{\partial x(t,\delta)}{\partial t} = F_\mu(x(t,\delta)), \qquad x(0,\delta) = x_0 + \delta g \tag{2.11}$$

The restriction in size of δ means that we are interested in the case of small δ. Then, we state a lemma which is a direct consequence of a result in [28]. The same result is also proved in §4, p. 40 [43], or as Theorem 3.1 in p. 95 of [30]. An alternative proof can be found in §2.5.7 of [2]:

Lemma 2.2.1 *The trajectory $x(t,\delta)$ is differentiable with respect to δ, and the vector partial derivative $\frac{\partial x(t,\delta)}{\partial \delta}(t,\delta)$ of $x(t,\delta)$ with respect to δ is continuous in δ.*

We denote this partial derivative $\frac{\partial x(t,\delta)}{\partial \delta}$ by $x_\delta(t,\delta)$.

We then show the following result:

Proposition 2.2.1 *If F_μ is continuously differentiable, then the directional derivative of J^c_μ in the direction of a vector $g \in \mathbf{R}^n$, $L_g J^c_\mu(x_0)$, exists and is continuous at any $x_0 \in X$.*

Proof:

Integration of (2.11) gives $x(t,\delta)$ as

$$x(t,\delta) = x_0 + \delta g + \int_0^t F_\mu(x(s,\delta))ds. \tag{2.12}$$

We differentiate (2.12) with respect to δ. Since F_μ is continuously differentiable, we can apply Leibniz's rule (Theorem 11.1, p. 282 in [51]) and obtain the derivative of the integral by differentiation of the integrand:

$$x_\delta(t,\delta) \triangleq \frac{\partial x(t,\delta)}{\partial \delta} = g + \int_0^t \left[\frac{\partial F_\mu(x(s,\delta))}{\partial x}\right] x_\delta(t,\delta)ds, \tag{2.13}$$

where by $\frac{\partial F_\mu}{\partial x}(x)$ we denote the $(n \times n)$-matrix whose ij-th element is given by

$$\left[\frac{\partial F}{\partial x}(x)\right]_{ij} = \frac{\partial F_i}{\partial x_j}(x). \tag{2.14}$$

The dynamics of x_δ are therefore given by

$$\frac{\partial x_\delta(t,\delta)}{\partial t} = \left[\frac{\partial F_\mu(x(t,\delta))}{\partial x}\right] x_\delta(t,\delta), \quad x_\delta(0,\delta) = g. \tag{2.15}$$

The quantity inside the brackets is the linearized dynamics of (2.1). We will show that in the limit $t \to \infty$, $x_\delta(t,\delta)$ converges to 0 exponentially. Since (2.1) is exponentially stable, it follows [76, 41] that its linearization at the origin

$$\dot{z} = \left[\frac{\partial F_\mu(0)}{\partial x}\right] z \tag{2.16}$$

is exponentially stable. We denote the matrix in (2.16) by A_0. It follows that

$$\|e^{A_0 t}\| \leq m e^{-\lambda t}, \tag{2.17}$$

for some positive constants m, λ. From exponential stability of (2.1) and continuous differentiability of F_μ it follows that for any given $\epsilon > 0$, we can find $T_\epsilon > 0$ such that

$$\left\| \left[\frac{\partial F_\mu(x(t,\delta))}{\partial x}\right] - A_0 \right\| < \epsilon \tag{2.18}$$

for every $t \in [T_\epsilon, \infty)$, where by $\|\cdot\|$ we denote both a norm on $\mathbf{R}^n$ and its induced matrix norm. Consider the sub-trajectory $x_\delta(t,\delta)$ for $t \in [T_\epsilon, \infty)$. According to Theorem 2, p. 36, in [7], applied to the linear time-varying system (2.15),

$$\|x_\delta(t,\delta)\| \leq m\|x_\delta(0,\delta)\| e^{-(\lambda - m\epsilon)t} = m\|g\| e^{-(\lambda - m\epsilon)t}, \qquad t \in [T_\epsilon, \infty). \tag{2.19}$$

We choose $\epsilon = \alpha\lambda/m$ for some $\alpha \in (0,1)$ in order to guarantee that this sub-trajectory is exponentially stable. It follows that $x_\delta(t,\delta)$ converges exponentially to 0 as $t \to 0$. This fact will prove useful in the sequel of the proof.

We define the function $J_f : \mathbf{R} \times \mathbf{R} \to \mathbf{R}$ given by

$$J_f(t,\delta) = \int_0^t (x(s,\delta)^T Q x(s,\delta) + \mu(x(s,\delta))^T R \mu(x(s,\delta)))ds \tag{2.20}$$

which converges to $J_\mu^c(x_0 + \delta g)$ as $t \to \infty$. The partial derivative of $J_f(t,\delta)$ with respect to δ is proven to exist and be continuous, for every t, in Section 2.4.2, p. 113, of [2]. We denote it by $J_{f\delta}(t,\delta)$. By taking derivatives of (2.20), and applying Leibniz's rule, we obtain:

$$J_{f\delta}(t,\delta) \triangleq \frac{\partial J_f(t,\delta)}{\partial \delta} = \int_0^t \left\{ \left[2x^T(s,\delta)Q + 2\mu(x(s,\delta))^T R \frac{\partial \mu(x(s,\delta))}{\partial x} \right] x_\delta(s,\delta) \right\} ds. \tag{2.21}$$

Since $x(s,\delta)$ is bounded in s and $|\delta| \leq 1$, and μ continuously differentiable, it follows that $\frac{\partial \mu(x(t,\delta))}{\partial x}$ is bounded. From continuity of μ, it follows that $\mu(x(t,\delta))$ is bounded, and therefore that $2\mu(x(t,\delta))^T R \frac{\partial \mu(x(t,\delta))}{\partial x}$ is bounded. In conjuction with the exponential convergence of $x_\delta(t,\delta)$, it follows that the integrand in (2.21) is summable over $[0,\infty)$. Let us confine δ in a bounded interval $[-\epsilon_\delta, \epsilon_\delta]$. Then, in conjuction with (2.19), it follows that the absolute value of the integrand in (2.21) is bounded by an expression

$$constant \cdot e^{-(\lambda - m\epsilon)t}, \qquad t \in [T_\epsilon, \infty), \qquad \delta \in [-\epsilon_\delta, \epsilon_\delta].$$

for some positive constant, and this expression is integrable between T_ϵ and ∞. Furthermore, the integrand of (2.21) is continuous. Therefore, we can apply Leibniz's rule for the tail of the integral (2.21) between T_ϵ and ∞ (Theorem 11.10, p.295 in [51]).

We can also apply Leibniz's rule between time 0 and T_ϵ. We then conclude that the limit

$$\lim_{t\to\infty} J_{f\delta}(t,\delta) \tag{2.22}$$

exists and is equal to the derivative $\frac{dJ^c_\mu(x_0+\delta g)}{d\delta}$, for $\delta \in [-\epsilon_\delta, \epsilon_\delta]$. At $\delta = 0$, this derivative is equal to $L_g J^c_\mu(x_0)$, that is, the directional derivative of J^c_μ in the direction g at the point x_0.
Finally, to prove continuity of $L_g J^c_\mu$ at x_0, we notice that the procedure followed for x_0 to obtain $J_{f\delta}(t,0)$ can be followed for any $y \in X$. We then denote the dependence on y as $J^y_{f\delta}(t,0)$ Then, we can define a function $J_m : \mathbf{R}^n \times \mathbf{R} \to \mathbf{R}$, where

$$J_m(y,t) \triangleq J^y_{f\delta}(t,0). \tag{2.23}$$

If $x_y(t,\delta)$ is the trajectory satisfying

$$\dot{x}_y(t,\delta) = F_\mu(x_y(t,\delta)), \qquad x_y(t=0,\delta) = y + \delta g, \tag{2.24}$$

then this function is given by (cf. (2.21)):

$$J_m(y,t) = \frac{\partial J^y_f(t,0)}{\partial\delta} = \int_0^t \{[2x^T_y(s,0)Q + \mu(x_y(s,0))^T R \frac{\partial\mu(x_y(s,0))}{\partial x_y}] \cdot x_{y\delta}(s,0)\}ds. \tag{2.25}$$

As time $t \to \infty$, $J_m(y,t)$ converges to $L_g J^c_\mu(y)$. Consider a real $\epsilon_1 > 0$. Then, there exists a real $T_c > 0$ such that:

$$||J_m(x_0,T_c) - L_g J^c_\mu(x_0)|| < \frac{\epsilon_1}{3}, \quad ||J_m(x,T_c) - L_g J^c_\mu(x_0)|| < \frac{\epsilon_1}{3} \tag{2.26}$$

Furthermore, from continuity of the dynamics in (2.24), it follows that the function $x_y(T_c,0)$ is continuous in y. Therefore, there exists a real $\epsilon_2 > 0$ such that, for all

$\|x - x_0\| < \epsilon_2$:

$$\|J_m(x_0, T_c) - J_m(x, T_c)\| < \frac{\epsilon_1}{3}. \tag{2.27}$$

By combining the last inequality with inequalities (2.26), it follows that

$$\|L_g J_\mu^c(x) - L_g J_\mu^c(x_0)\| < \epsilon_1,$$

for every $\|x - x_0\| < \epsilon_2$. ■

2.3 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 μ'. As we saw in Chapter 1, we approximate J_μ^c by $\tilde{J}_\mu$ and, recalling definition (1.6) and (1.17), the new controller is given by

$$\begin{aligned} \mu'(x) = [\mu_1'(x), \cdots, \mu_m'(x)]^T = & \ -\tfrac{1}{2} R^{-1} L_G(x) \tilde{J}_\mu(x) \\ = & \ -\tfrac{1}{2} R^{-1} [L_{g_1(x)} \tilde{J}_\mu(x), \cdots, L_{g_m(x)} \tilde{J}_\mu(x)]^T. \end{aligned} \tag{2.28}$$

Unless $\tilde{J}_\mu$ is a "good" approximation of J_μ^c, the resulting controller, μ' 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 μ' is guaranteed. A thorough discussion of the result is given in Section 2.4. Before stating the theorem, note the input weight matrix R can be written in the form $R = R_1^T R_1$, where R_1 is some square nonsingular matrix, by virtue of being a symmetric positive definite matrix (cf. Theorem 8.14 in [22]).

Theorem 2.3.1 *Assume that f, G, μ and $\tilde{J}_\mu$ are continuously differentiable. Let the square nonsingular matrix R_1 be such that $R = R_1^T R_1$. If*

$$-x^T Qx - \frac{1}{2}\mu(x)^T R\mu(x) - \frac{1}{2}[R_1\mu(x) + R_1^{-1}L_G J_\mu^c(x)]^T[R_1\mu(x) + R_1^{-1}L_G J_\mu^c(x)]$$
$$+\frac{1}{2}L_G J_\mu^c(x)^T R^{-1}[L_G J_\mu^c(x) - L_G\tilde{J}_\mu(x)] \leq -\zeta\|x\|^2, \tag{2.29}$$

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

$$\dot{x} = f(x) + G(x)\mu'(x), \tag{2.30}$$

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

Proof: We will show that J_μ^c is a Lyapunov function for the system described by (2.30), provided that (2.29) is satisfied. Since 0 is an equilibrium point of (2.1), it follows that $J_\mu^c(0) = 0$. Obviously, J_μ^c is strictly positive in the rest of X. Before we consider the decay rate of J_μ^c along the trajectories of the system described by (2.30), we obtain a valuable relation by examining its rate of change along the trajectories of the system described by (2.1). By continuous differentiability of J_μ^c and the chain rule, it follows that the rate of change of J_μ^c along the trajectories of the system described by (2.1) is given by

$$\frac{d}{dt}J_\mu^c(x(t)) = \nabla J_\mu^c(x(t)) \cdot (f(x(t)) + G(x(t))\mu(x(t))) = L_{f+G\mu}J_\mu^c(x(t)). \tag{2.31}$$

Since J_μ^c is continuously differentiable (Proposition 2.2.1), it follows that its directional derivatives are linear with respect to the direction vector (Lemma (4.1.5), p. 255, of [24]), that is,

$$L_{(f+G\mu)}J_\mu^c(x(t)) \;\; = \;\; L_f J_\mu^c(x(t)) + L_{G\mu}J_\mu^c(x(t))^T$$

$$\begin{aligned} &= L_f J_\mu^c(x(t)) + L_G J_\mu^c(x(t))^T \cdot \mu(x) \\ &= L_f J_\mu^c(x(t)) + L_{g_1} J_\mu^c(x(t)) \cdot \mu_1(x) + \ldots + \\ &\quad + L_{g_m} J_\mu^c(x(t)) \cdot \mu_m(x), \end{aligned} \tag{2.32}$$

where $x(t)$ denotes a trajectory of (2.1). Existence of the directional derivatives was established in Proposition 2.2.1. Furthermore,

$$J_\mu^c(x(t)) = \int_t^\infty \left(x(\tau)^T Q x(\tau) + \mu(x(\tau))^T R \mu(x(\tau)) \right) d\tau.$$

Therefore, differentiation with respect to t leads to

$$\frac{d}{dt} J_\mu^c(x(t)) = -x(t)^T Q x(t) - \mu(x(t))^T R \mu(x(t)) \tag{2.33}$$

We combine (2.31) with (2.32) and (2.33) to obtain

$$L_f J_\mu^c(x) + L_G J_\mu^c(x)^T \mu(x) = -x^T Q x - \mu(x)^T R \mu(x). \tag{2.34}$$

Note that (2.34) holds for every x in X, since every x is part of at least one possible trajectory of the system (that is, the trajectory which starts off at x at time $t = 0$). Using the form (2.28) of μ', the rate of change of J_μ^c along the trajectories of (2.30) is given, in a similar manner to (2.32), by

$$\begin{aligned} L_{(f+G\mu')} J_\mu^c(x(t)) &= L_f J_\mu^c(x(t)) + L_G J_\mu^c(x(t))^T \mu'(x(t)) \\ &= L_f J_\mu^c(x(t)) - \frac{1}{2} L_G J_\mu^c(x(t))^T R^{-1} L_G \tilde{J}_\mu(x(t)) \end{aligned} \tag{2.35}$$

Since every x is part of a possible trajectory of (2.30), the dependence on time in (2.35) can be omitted. We solve (2.34) for $L_f J^c_\mu(x)$ and replace in (2.35), thus obtaining

$$L_{(f+G\mu')} J^c_\mu(x) = -x^T Q x - \mu(x)^T R \mu(x) - L_G J^c_\mu(x)^T \mu(x) - \frac{1}{2} L_G J^c_\mu(x(t))^T R^{-1} L_G \tilde{J}_\mu(x) \tag{2.36}$$

Splitting $-\mu(x)^T R\mu(x)$ into $-\frac{1}{2}\mu(x)^T R\mu(x) - \frac{1}{2}\mu(x)^T R\mu(x)$ and adding and subtracting $\frac{1}{2} L_G J^c_\mu(x)^T R^{-1} L_G J^c_\mu(x)$ in the right hand side of (2.36) proves that

$$\begin{aligned} L_{(f+G\mu')} J^c_\mu(x) = & \quad -x^T Q x - \tfrac{1}{2}\mu(x)^T R \mu(x) \\ & -\tfrac{1}{2}[R_1 \mu(x) + R_1^{-1} L_G J^c_\mu(x)]^T [R_1 \mu(x) + R_1^{-1} L_G J^c_\mu(x)] \\ & +\tfrac{1}{2} L_G J^c_\mu(x)^T R^{-1} [L_G J^c_\mu(x) - L_G \tilde{J}_\mu(x)]. \end{aligned} \tag{2.37}$$

Clearly, if (2.29) is satisfied, then

$$L_{(f+G\mu')} J^c_\mu(x) < -\zeta \|x\|^2 \tag{2.38}$$

and this implies that J^c_μ is a Lyapunov function for (2.30). The function J^c_μ is a Lyapunov function for the closed loop system under μ and μ'. In conjuction with Assumption (2.1.5), it turns out that all trajectories of the closed loop system under μ' starting inside X_0 never leave X, since (2.38) holds throughout X. The last inequality (2.38), in conjuction with (2.9), also implies that (2.30) is exponentially stable, according to Corollary 3.4, p.140, in [41]. ■

2.4 Discussion and interpretation of the stability criterion

In the previous section, we developed a condition (2.29) on the difference between the directional derivatives of the *actual* cost function of (2.1) and of the *approximate* function $\tilde{J}_\mu$, along the directions of the input vectors $g_i(x)$. Condition (2.29) suffices to guarantee exponential stability of the policy μ'. It is notable that the sufficient criterion only involves these directional derivatives, and not the actual difference between the functions. This is a direct consequence of the very nature of the dynamic system and optimal control problem. The dynamics and cost function are differentiable in the state. This enables the first order Taylor expansions, (1.7) and (1.15), which not only result in algorithm simplifications, but also in the stability condition. Notably, the Taylor expansion allowed us to obtain a closed form expression for μ'. We construct $\tilde{J}_\mu$ based on samples of the discrete time system's cost function. The working assumption here is that the discretization interval δ is small enough such that J^c_μ and J^d_μ are close. It is the designer's responsibility to select, given a dynamic system, a small enough δ such that the error introduced by discretization is "small"; "small", in the sense that the approximation $\tilde{J}_\mu$, which is tuned via the samples of J^d_μ, satisfies (2.29). Since policy evaluation is performed off-line, the designer is not constrained (at least in theory) about the size of δ that he/she wishes to use.

Consider the case where there is no approximation error in the directional derivative. That is, assume that $L_G\tilde{J}_\mu(x) = L_G J^c_\mu(x)$ for every x. In that case, we expect that the policy μ' is exponentially stabilizing. Indeed, the left hand side of (2.29) is a sum of negative terms,

$$-x^TQx - \frac{1}{2}\mu(x)^TR\mu(x) - \frac{1}{2}[R_1\mu(x) + R_1^{-1}L_GJ^c_\mu(x)]^T[R_1\mu(x) + R_1^{-1}L_GJ^c_\mu(x)], \quad (2.39)$$

which is less than or equal to $-\frac{k_1}{2}\|x\|^2$ according to Lemma 2.1.2, and thus J_μ^c is an exponential Lyapunov function for (2.30). Furthermore, it is obvious that there exists at least a $\frac{k_1}{2}\|x\|^2$ positive margin that can be tolerated. Therefore, condition (2.29) does allow a nonzero approximation error despite which the updated controller μ' is stabilizing.

Let us discuss some aspects of the criterion, for the single input case, for simplicity. We observe the following fact:

Remark 2.4.1 *Suppose that the directional derivative $L_g\tilde{J}_\mu(x)$ has the same sign as $L_gJ_\mu^c(x)$, and*

$$|L_g\tilde{J}_\mu(x)| > |L_gJ_\mu^c(x)|. \tag{2.40}$$

It follows that

$$L_gJ_\mu^c(x)\left(L_gJ_\mu^c(x) - L_g\tilde{J}_\mu(x)\right) \leq 0, \tag{2.41}$$

which in turn implies that the sufficient condition (2.29) is satisfied at x.

Indeed, if $L_gJ_\mu^c(x) \geq 0$, and (2.40) holds, then $[L_gJ_\mu^c(x) - L_g\tilde{J}_\mu(x)] \leq 0$, and (2.41) follows. Similarly, if $L_gJ_\mu^c(x) \leq 0$, and (2.40) holds, then $[L_gJ_\mu^c(x) - L_g\tilde{J}_\mu(x)] \geq 0$, and (2.41) again follows. This shows that if the assumptions of Remark 2.4.1 hold, then (2.29) 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 μ_e generated via policy iteration over μ, assuming zero cost function approximation error, that is,

$$\mu_e(x) = -\frac{1}{2}R^{-1}L_gJ_\mu^c(x). \tag{2.42}$$

As we saw above, J_μ^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_μ^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_μ^c strictly decreases. According to Remark 2.4.1, 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_μ^c.

In the case that the drift dynamics $f(x)$ point towards the direction of increasing J_μ^c, tending to destabilize the system, the product $g(x)\mu_e(x)$ has to point towards decreasing values of J_μ^c, or else their sum would not. Then, it is easy to see that any $g(x)\mu_m(x)$, with $\mu_m(x)$ as above, points even more strongly towards the direction of decreasing J_μ^c. So does the vector sum.

However, even in the opposite case that $f(x)$ points towards decreasing J_μ^c, we can conclude that $g(x)\mu_e(x)$ does so too! For if it did not, there would be some $\mu_m(x)$, of the same sign as $\mu_e(x)$ and of large enough magnitude, such that the vector sum of $f(x)$ and $g(x)\mu_m(x)$ points towards increasing J_μ^c. This is a contradiction based on Theorem 2.3.1.

This is a useful conclusion, because it demonstrates that *exact* policy iteration generates a policy μ_e which always pushes towards the direction of decreasing J_μ^c, thus trying to minimize cost accumulation. Cost minimization via a single policy iteration, clearly amounts to decreasing values of J_μ^c along any trajectory of the closed loop system. This possibly explains why this function gave us such a reasonable criterion, and suggests that it would not be easy to obtain a less conservative criterion by any other means.

Finally, similar arguments to the above can be made for the multi-input case where $u \in \mathbf{R}^m$. In this case, each input u_i, $i = 1, \ldots, m$, multiplies the i-th column $g_i(x)$ of the input matrix $G(x)$, and the above arguments hold for each pair of input u_i and the corresponding input vector $g_i(x)$.

2.5 Sufficient Condition for Cost Improvement of a single Iteration

We consider the case in which a policy μ is given such that $J^c_\mu(x) < \infty$, and we perform an approximate policy iteration which results in a policy μ' given by (2.28). In this section, we develop conditions on the approximation error, such that the cost function $J^c_{\mu'}$ of the closed loop system under policy μ' is smaller than J^c_μ. We now state the result:

Theorem 2.5.1 *Let f, G, μ be continuously differentiable, and let $\tilde{J}_\mu$ be selected twice continuously differentiable, so that μ' is continuously differentiable. Assume that the error in approximating J^c_μ by $\tilde{J}_\mu$ is small enough such that the closed loop system under μ' is exponentially stable, and $J^c_{\mu'}(x) < \infty$ for all $x \in X$. If*

$$\left(\mu'(x)^T R \mu'(x) - \mu(x)^T R \mu(x)\right) + L_G J^c_\mu(x)^T \left(\mu'(x) - \mu(x)\right) \leq 0 \tag{2.43}$$

for $x \in X$, then the performance of the closed loop system under μ' is improved compared to the performance of the closed loop system under μ with respect to the cost functional (1.3), that is,

$$J^c_{\mu'}(x) \leq J^c_\mu(x), \tag{2.44}$$

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 (2.43) takes the form

$$\left(\frac{1}{4}L_G\tilde{J}_\mu(x)^T R^{-1} L_G\tilde{J}_\mu(x) - \mu(x)^T R\mu(x)\right) + L_G J^c_\mu(x)^T \left(-\frac{1}{2}R^{-1}L_G\tilde{J}_\mu(x) - \mu(x)\right) \leq 0, \tag{2.45}$$

for all $x \in X$.

Proof:

We rewrite equality (2.34),

$$L_f J_\mu^c(x) + L_G J_\mu^c(x)^T \mu(x) = -x^T Q x - \mu(x)^T R \mu(x), \qquad \text{for every } x \in X, \quad (2.46)$$

which expresses the rate of change of the cost function J_μ^c along a trajectory of the closed loop system under μ as the negative integrand of the integral that defines J_μ^c. Equality (2.46) is used later in the proof.

We consider any point x_0 in X, which we fix for the rest of the proof. We denote by $y(t)$ the trajectory of the closed loop system under policy μ' which starts off at x_0 at time $t = 0$, that is,

$$\dot{y}(t) = f(y(t)) + G(y(t))\mu'(y(t)), \quad y(0) = x_0. \quad (2.47)$$

We define the function $I(t)$ as:

$$I(t) \triangleq L_f J_\mu^c(y(t)) + L_G J_\mu^c(y(t))^T \mu'(y(t)) + y(t)^T Q y(t) + \mu'(y(t))^T R \mu'(y(t)). \quad (2.48)$$

Consider the right hand side of the above definition. The sum of its first two terms is equal to the rate of change of J_μ^c along a trajectory of the closed loop system under μ'. The sum of the last two terms is the integrand of the integral that defines $J_{\mu'}^c(x_0)$. Therefore, by integrating $I(t)$ between time 0 and ∞, we obtain:

$$\begin{aligned}
\int_0^\infty I(t)dt = & \int_0^\infty \left(L_f J_\mu^c(y(t)) + L_G J_\mu^c(y(t))^T \mu'(y(t))\right) dt \\
& + \int_0^\infty \left(y(t)^T Q y(t) + \mu'(y(t))^T R \mu'(y(t))\right) dt \\
= & \int_0^\infty \frac{dJ_\mu^c(y(t))}{dt} dt + J_{\mu'}^c(x_0) \\
= & \lim_{t\to\infty} J_\mu^c(y(t)) - J_\mu^c(x_0) + J_{\mu'}^c(x_0)
\end{aligned}$$

$$= \qquad J^c_{\mu'}(x_0) - J^c_\mu(x_0), \tag{2.49}$$

where $\lim_{t\to\infty} J^c_\mu(y(t)) = 0$ by virtue of exponential stability of the closed loop system under μ' and $J^c_\mu(0) = 0$.

By adding and subtracting $L_G J^c_\mu(y(t))^T \mu(y(t))$ and $\mu(y(t))^T R\mu(y(t))$ in (2.48), and then by rearranging the terms, $I(t)$ is written as:

$$\begin{aligned}
I(t) = \quad & L_f J^c_\mu(y(t)) + L_G J^c_\mu(y(t))^T \mu'(y(t)) + L_G J^c_\mu(y(t))^T \mu(y(t)) \\
& -L_G J^c_\mu(y(t))^T \mu(y(t)) + y(t)^T Q y(t) + \mu'(y(t))^T R \mu'(y(t)) \\
& +\mu(y(t))^T R\mu(y(t)) - \mu(y(t))^T R\mu(y(t)) \\
= \quad & L_f J^c_\mu(y(t)) + L_G J^c_\mu(y(t))^T \mu(y(t)) + y(t)^T Q y(t) + \mu(y(t))^T R\mu(y(t)) + \\
& +L_G J^c_\mu(y(t))^T \left(\mu'(y(t)) - \mu(y(t))\right) \\
& + \Big(\mu'(y(t))^T R\mu'(y(t)) - \mu(y(t))^T R\mu(y(t))\Big)
\end{aligned} \tag{2.50}$$

By virtue of (2.46), the sum of the first four terms on the right side of the last equality is equal to 0. It then follows from (2.43) that

$$\begin{aligned}
I(t) = \; & \Big(\mu'(y(t))^T R\mu'(y(t)) - \mu(y(t))^T R\mu(y(t))\Big) \\
& +L_G J^c_\mu(y(t))^T \left(\mu'(y(t)) - \mu(y(t))\right) \\
\le \; & 0.
\end{aligned} \tag{2.51}$$

It follows that the integral of $I(t)$ is nonpositive. Therefore, (2.49) implies that

$$J^c_{\mu'}(x_0) \le J^c_\mu(x_0). \tag{2.52}$$

This completes the proof, since (2.52) holds for any x_0 in X. ■

2.6 Discussion of the cost improvement result

To see whether inequality (2.45) 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, the left hand side of inequality (2.45) becomes:

$$-\frac{1}{4} L_G J^c_\mu(x)^T R^{-1} L_G J^c_\mu(x) - \mu(x)^T R \mu(x) - L_G J^c_\mu(x)^T \mu(x) =$$
$$= -\left[\frac{1}{2} R_1^{-1} L_G J^c_\mu(x) + R_1 \mu(x)\right]^T \left[\frac{1}{2} R_1^{-1} L_G J^c_\mu(x) + R_1 \mu(x)\right], \tag{2.53}$$

which is clearly less than or equal to 0. Thus, inequality (2.45) 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 2.5.1 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_μ, that is, $L_g J^c_\mu(x) < 0$. The improvement criterion (2.43) for an updated policy μ' is satisfied if

$$\mu'(x)^2 + L_g J^c_\mu(x) \mu'(x) = \mu'(x) \left(\mu'(x) + L_g J^c_\mu(x)\right) \leq 0 \tag{2.54}$$

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_μ^c. Therefore, the rate of change of J_μ^c along trajectories of the closed loop system under the improving controller μ' is faster than in the open loop case. However, μ' 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.

Chapter 3

Upper Bound on Suboptimality as Iterations Repeat

It now is assumed that the control designer manages to generate a sequence $\mu_1, \mu_2, \ldots, \mu_k, \ldots$ of exponentially stable policies. In Chapter 2, we provided a sufficient condition on the approximation error that guarantee exponential stability of each of these policies, and a sufficient condition that guarantee improvement after a single iteration. Nevertheless, in practice, the controllers that are generated are suboptimal. Our objective in this chapter is to establish an upper bound on the suboptimality of the controllers that are generated after a large number of iterations. A meaningful upper bound has to be a monotonically increasing function of the approximation error, and vanish when the approximation error vanishes. For practical purposes, these worst case bounds might prove quite conservative. However, they are of theoretical value as well as of conceptual importance for the designer, since they indicate what improvement or degradation he/she should expect by tightening or relaxing his/her efforts at the stage of cost function approximation. Similar suboptimality bounds have only recently been given in the cases of discounted cost problems or stochastic shortest path problems for systems with countable state spaces. These results were developed by Bertsekas and Tsitsiklis in Chapter 6 of [11]. The work in this chapter draws from their ideas and extends them to the undiscounted problem for systems with euclidean state spaces, which constitute the subject of this thesis.

In this chapter, for simplicity of notation, we assume that the control input u is scalar. For the same reason, we consider the case where the state and input weight matrices are $Q = I$, $R = 1$. The general case follows with slight modifications.

Preview:

We consider the case where we implement approximate policy iteration as described in Chapter 1, with a discretization interval δ. The policy update rule is given by (1.17). We assume that the absolute value error between the directional derivative of the approximate cost function $L_g\tilde{J}_{\mu_k}(x)$ and the directional derivative $L_g J^d_{\mu_k}(x)$ of the discretized system cost function is upper bounded by a function $\epsilon(x)$ over all k. We derive an upper bound on suboptimality of the generated policies. The upper bound is achieved after a certain number of iterations. In the case where $\epsilon(x)$ decreases to 0 uniformly over x, the upper bound collapses to a minimum. This minimum is in general nonzero, even in the case where we allow the discretization interval δ to asymptotically decrease to 0. At this point, it is unclear whether the upper bound can be made to collapse to zero with a different line of proof, or with a somewhat different set of assumptions.

3.1 Error Definitions

As was discussed in Chapter 1, we employ approximate policy iteration in order to design a suboptimal controller/policy for the problem (1.7), (1.9), which constitutes a discrete time representation of the original, continuous time problem, (1.1), (1.3). The sampling time, δ, is fixed throughout the policy iteration process. In this chapter, we omit the superscript ^{d} in the symbol that we use to denote the *actual* cost function of

$$x_{t+1} = x_t + \delta(f(x_t) + g(x_t)u_t), \qquad x_{t=0} = x_0. \tag{3.1}$$

under a policy $u_t = \mu_k(x_t)$. The actual cost function of this discrete time system is thus denoted by J_{μ_k}, and $J_{\mu_k}(x_0)$ represents the cost

$$\sum_{t=0}^{\infty} \delta(x_t^T Q x_t + u_t^T R u_t), \tag{3.2}$$

accumulated along the trajectory starting at x_0, under control $u_t = \mu_k(x_t)$. The *approximate* cost function is denoted by $\tilde{J}_{\mu_k}$. It is assumed that the vectors f, g are differentiable. The same is assumed of each policy μ_k, $k = 1, 2, \ldots$. For the latter, it suffices that the initial policy μ_0 is differentiable and that our choice of $\tilde{J}_{\mu_k}$ is twice differentiable, by virtue of (1.17). The following proposition can then be shown:

Proposition 3.1.1 *Assuming that f, g, μ_k are continuously differentiable and the system (3.1) controlled by policy μ_k is exponentially stable, then the actual cost function J_{μ_k} of (3.1) is continuously differentiable.*

By virtue of this proposition, it makes sense to use differentiable approximator functions $\tilde{J}_{\mu_k}$ and the policy update rule (1.17). Nevertheless, by using (1.17) we introduce a policy-update-error, which is induced by the approximation (1.15). We will take a closer look at this. First, we introduce some *shorthand notation*, which will be used throughout the chapter. For any function $J : \mathbf{R}^n \to \mathbf{R}$, we consider the function $TJ : \mathbf{R}^n \to \mathbf{R}$ obtained by applying one exact discrete-time dynamic programming iteration as follows:

$$(TJ)(x) = \min_u \{\delta(x^T x + u^2) + J(x + \delta(f(x) + g(x)u))\} \tag{3.3}$$

Similarly, for any function $J : \mathbf{R}^n \to \mathbf{R}$ and policy μ we consider the function $T_\mu J : \mathbf{R}^n \to \mathbf{R}$ defined as

$$(T_\mu J)(x) = \delta(x^T x + \mu(x)^2) + J(x + \delta(f(x) + g(x)\mu(x)))\}. \tag{3.4}$$

Given a policy μ_k and the corresponding approximate cost function $\tilde{J}_{\mu_k}$, let us denote by μ^e_{k+1} the policy resulting from the *exact* policy update

$$\mu^e_{k+1}(x) = \arg T\tilde{J}_{\mu_k}(x) = \arg\min_u\{\delta(x^T x + u^2) + \tilde{J}_{\mu_k}(x + \delta(f(x) + g(x)u))\}. \tag{3.5}$$

By applying Taylor's theorem (cf. (3.6.2) in p. 198 of [24]), $\tilde{J}_{\mu_k}(x + \delta(f(x) + g(x)u))$ is written as

$$\tilde{J}_{\mu_k}(x)+\delta L_{(f(x)+g(x)u)}\tilde{J}_{\mu_k}(x)+\tilde{\zeta}(x,u) = \tilde{J}_{\mu_k}(x)+\delta(L_{f(x)}\tilde{J}_{\mu_k}(x)+u\cdot L_{g(x)}\tilde{J}_{\mu_k}(x))+\tilde{\zeta}(x,u) \tag{3.6}$$

where $\tilde{\zeta}(x,u)$ is of order $o(\delta)$, as $\delta \to 0$, for any fixed x and u. However, we actually implement policy updating according to (1.16) which, in contrast to (3.5), is a convex optimization problem with a closed form solution (1.17). Thus,

$$\mu_{k+1} = \arg\min_u\{\delta u^2 + \delta L_g \tilde{J}_{\mu_k} \cdot u\} \tag{3.7}$$

The function $\tilde{\zeta}(x, \mu^e_{k+1}(x))$ is denoted by $\tilde{\zeta}_1(x)$. It follows that

$$T\tilde{J}_{\mu_k} = T_{\mu^e_{k+1}}\tilde{J}_{\mu_k} = \delta x^T x + \delta(\mu^e_{k+1})^2 + \tilde{J}_{\mu_k} + \delta L_f \tilde{J}_{\mu_k} + \delta L_g \tilde{J}_{\mu_k} \cdot \mu^e_{k+1} + \tilde{\zeta}_1. \tag{3.8}$$

We wish to characterize the difference between $T_{\mu_{k+1}}\tilde{J}_{\mu_k}$ and $T\tilde{J}_{\mu_k}$. From the definition of $T_{\mu_{k+1}}\tilde{J}_{\mu_k}$ and differentiability of $\tilde{J}_{\mu_k}$, we obtain

$$T_{\mu_{k+1}}\tilde{J}_{\mu_k} = \delta x^T x + \delta\mu_{k+1}(x)^2 + \tilde{J}_{\mu_k}(x) + \delta L_f \tilde{J}_{\mu_k} + \delta L_g \tilde{J}_{\mu_k} \cdot \mu_{k+1} + \tilde{\zeta}_2(x), \qquad (3.9)$$

where $\tilde{\zeta}_2(x) \triangleq \tilde{\zeta}(x, \mu_{k+1}(x))$ is of order $o(\delta)$ as $\delta \to 0$ (for any fixed x). We define a new function $\tilde{\xi}$, which will be used in the sequel, as

$$\tilde{\xi}(x) \triangleq \delta\mu_{k+1}(x)^2 + \delta L_g \tilde{J}_{\mu_k}(x) \cdot \mu_{k+1}(x) - \delta\mu_{k+1}^e(x)^2 - \delta L_g \tilde{J}_{\mu_k}(x) \cdot \mu_{k+1}^e(x) \qquad (3.10)$$

By subtracting (3.9) from (3.8) and using the definition (3.10), we obtain

$$T_{\mu_{k+1}}\tilde{J}_{\mu_k}(x) - T\tilde{J}_{\mu_k}(x) = \tilde{\xi}(x) + \tilde{\zeta}_2(x) - \tilde{\zeta}_1(x) \geq 0. \qquad (3.11)$$

The above difference is non-negative by virtue of the fact that $T\tilde{J}_{\mu_k}(x)$ minimizes the expression in (3.3). At the same time, $\tilde{\xi}$ is non-positive, by virtue of μ_{k+1} minimizing the expression in (3.7). Therefore, the difference $\tilde{\zeta}_2(x) - \tilde{\zeta}_1(x)$ is non-negative. From these arguments, it follows that $\tilde{\xi}(x)$ is smaller in absolute value than $\tilde{\zeta}_2(x) - \tilde{\zeta}_1(x)$ for every x. It follows that $\tilde{\xi}(x)$ is of order $o(\delta)$ as $\delta \to 0$, for any fixed x. The sum $\tilde{\xi}(x) + \tilde{\zeta}_2(x) - \tilde{\zeta}_1(x)$ represents the error introduced in the policy update step. For small δ, this error is negligible.

Similar expansions based on Taylor's theorem can be applied to $TJ_{\mu_k}(x)$ and $T_{\mu_{k+1}}J_{\mu_k}(x)$. By applying Taylor's theorem on J_{μ_k}, we obtain

$$J_{\mu_k}(x + \delta(f(x) + g(x)u) = J_{\mu_k}(x) + \delta(L_{f(x)}J_{\mu_k}(x) + u \cdot L_{g(x)}J_{\mu_k}(x)) + \zeta(x,u), \quad (3.12)$$

where $\zeta(x,u)$ is of order $o(\delta)$, as $\delta \to 0$, for any fixed x and u. We define the policies ν_{k+1}^e and ν_{k+1} resulting from J_{μ_k}:

$$\nu_{k+1}^e(x) = \arg TJ_{\mu_k}(x) = \arg\min_u\{\delta(x^Tx+u^2) + J_{\mu_k}(x+\delta(f(x)+g(x)u))\}, \quad (3.13)$$

that is, ν_{k+1}^e is the policy obtained by a single *exact* policy iteration step from μ_k. We also define

$$\nu_{k+1}(x) = \arg\min_u\{\delta u^2 + L_gJ_{\mu_k}(x)u\} = -\frac{1}{2}L_gJ_{\mu_k}(x) \quad (3.14)$$

We also define the following functions:

$$\zeta_1(x) \triangleq \zeta(x, \nu_{k+1}^e(x)) \quad (3.15)$$

$$\zeta_2(x) \triangleq \zeta(x, \nu_{k+1}(x)) \quad (3.16)$$

$$\zeta_3(x) \triangleq \zeta(x, \mu_{k+1}(x)) \quad (3.17)$$

$$\xi(x) \triangleq \delta\nu_{k+1}(x)^2 + \delta L_gJ_{\mu_k}(x)\cdot\nu_{k+1}(x) - \delta\nu_{k+1}^e(x)^2 - \delta L_gJ_{\mu_k}(x)\cdot\nu_{k+1}^e(x) \quad (3.18)$$

In a similar fashion to (3.11), we have

$$T_{\nu_{k+1}}J_{\mu_k}(x) = TJ_{\mu_k}(x) + \xi(x) + \zeta_2(x) - \zeta_1(x) \geq 0. \quad (3.19)$$

It can be shown that $\xi(x)$ is of order $o(\delta)$, as $\delta \to 0$ (for any fixed x), similarly to $\tilde{\xi}(x)$. We also define the function η as

$$\eta(x) \triangleq \xi(x) + \zeta_3(x) - \zeta_1(x), \quad (3.20)$$

which is of order $o(\delta)$, as $\delta \to 0$, for any fixed x.

The *optimal* policy which minimizes the cost (3.2) for the system (3.1) is denoted by μ^* and the optimal cost function by J^*. No differentiability guarantees can be established for either.

Some more notation needs to be introduced. First, we introduce the following notation for the approximation error between $L_g\tilde{J}_{\mu_k}$ and $L_gJ_{\mu_k}$:

$$\epsilon_k(x) \triangleq (L_{g(x)}\tilde{J}_{\mu_k}(x) - L_{g(x)}J_{\mu_k}(x))^2. \tag{3.21}$$

We also introduce a more compact notation for the dynamics of (3.1):

$$F_k(x) \triangleq x + \delta(f(x) + g(x)\mu_k(x)). \tag{3.22}$$

Consider a trajectory of (3.1) that starts at x. We use the notation $F_k^i(x)$ to denote the operator that maps x to the point reached by the system after i–steps along this trajectory. That is, $F_k^0(x) = x$, $F_k^1(x) = F_k(x)$, $F_k^2(x) = F_k(F_k(x))$, and, in general:

$$F_k^i(x) = \underbrace{F_k(F_k(\cdots F_k(}_{i\text{–times}}x)\cdots)) \tag{3.23}$$

Similarly, we may define the operator $F_*^i(x)$ for the system under optimal control, μ^*. Since all policies $\mu_0, \mu_1, \cdots$ and μ^* are exponentially stable, all trajectories of the discrete time dynamic system under these policies converge to the origin. Any ball around the origin, no matter how small it is, will be reached by a trajectory of the system in finite time. At the same time, the cost function J_{μ_k} decreases continuously along the trajectory and converges to 0 as the trajectory converges to the origin. Therefore, for every $\xi > 0$ we can define:

$$M_k(x,\xi) \triangleq \min\{i\ ; \quad |J_{\mu_k}(F_{k+1}^i(x))| < \frac{\xi}{2} \text{ and } |J_{\mu_{k+1}}(F_{k+1}^i(x))| < \frac{\xi}{2}\} \tag{3.24}$$

$$N_k(x,\xi) \triangleq \min\{i\ ; \quad |J_{\mu_{k+1}}(F_*^i(x))| < \frac{\xi}{2} \text{ and } |J^*(F_*^i(x))| < \frac{\xi}{2}\} \tag{3.25}$$

We assume that the following bounds exist and are finite:

$$\epsilon(x) \triangleq \sup_k \epsilon_k(x) \tag{3.26}$$

$$M(x,\xi) \triangleq \sup_k M_k(x,\xi) \tag{3.27}$$

$$N(x,\xi) \triangleq \sup_k N_k(x,\xi) \tag{3.28}$$

If we keep applying approximate policy iteration indefinitely, and thereby generate an infinite sequence of exponentially stable policies, there is a certain possibility that the above suprema do not exist. At any time during the policy iteration process, there exists no way to guarantee that no future policy will result in longer transients than the already generated policies. Therefore, finiteness of the bounds $\epsilon(x)$, $M(x,\xi)$ and $N(x,\xi)$ is not guaranteed and we have to assume it. Before stating the result, we also need to make the definitions that follow:

$$n(x) \triangleq N(x,\epsilon(x)) \tag{3.29}$$

$$H_i(x) \triangleq F^1_{k+1-i}(F^i_*(x)) \tag{3.30}$$

$$m_i(x) \triangleq M(H_i(x), \frac{\delta}{4}\epsilon(H_i(x)), \tag{3.31}$$

where δ is the discretization interval used in (1.7). The above quantities appear in the upper bound given below.

A last reminder before the result. We use extensively the directional derivative function L_gJ. In the following upper bound, as well as throughout the proof that follows, by $L_gJ(x)$, we denote the expression $L_{g(x)}(J(x))$

3.2 The Upper Bound

We now formulate the main result of this chapter:

Theorem 3.2.1 *Let the policy approximation errors $\epsilon_k(x)$ at the k-th iteration. Consider the definitions of the quantities $\xi(x)$, $\zeta_1(x)$, $\zeta_2(x)$, $\tilde{\xi}(x)$, $\tilde{\zeta}_1(x)$, $\tilde{\zeta}_2(x)$ which result from adopting the policy update rule (3.7) in place of the exact formula (3.5). Then, an upper bound on suboptimality of the policies μ_k for $k \geq n(x)$, where $n(x)$ is defined in (3.29), is given by*

$$\begin{aligned} J_{\mu_k}(x) - J^*(x) \leq \quad & \epsilon(x) + \tfrac{\delta}{4} \textstyle\sum_{i=0}^{n(x)-1} \epsilon(F_*^i(x)) + \sum_{i=0}^{n(x)-1} \eta(F_*^i(x)) + \\ & + \textstyle\sum_{i=0}^{n(x)-1} \{ \tfrac{\delta}{4}\epsilon(H_i(x)) + \tfrac{\delta}{4} \sum_{j=0}^{m_i(x)} \epsilon(F_{k-i}^j(H_i(x))) + \\ & + \textstyle\sum_{j=0}^{m_i(x)} \eta(F_{k-i}^j(H_i(x)))\} \end{aligned} \tag{3.32}$$

Before giving the proof of the theorem, we make some remarks on the upper bound. The function $\epsilon(x)$ represents the cost function approximation error at x, whereas $\eta(s)$ is a measure of the errors introduced by the Taylor based simplifications we have made in the policy update step. In the sequel, we shall try to interpret each term of the bound. In particular, we examine how the upper bound behaves as the approximation error on the directional derivative of the cost function uniformly goes to 0, as well as when the discretization constant δ goes to 0.

The first term, $\epsilon(x)$ clearly vanishes as the approximation error on the directional derivative of the cost function vanishes uniformly. Let us consider the second term, $\frac{\delta}{4}\sum_{i=0}^{n(x)-1} \epsilon(F_*^i(x))$. The sum is taken along a trajectory of the closed loop system under optimal control. It clearly vanishes as the approximation error goes to 0. We now consider how the sum behaves as the discretization constant δ goes to 0. Clearly in that case, $n(x)$ increases roughly as $1/\delta$. However, this does not imply that the term increases as δ decreases. Indeed, the sum is multiplied by δ. Therefore, it converges, as $\delta \to 0$, to the integral of the error $\epsilon(x)$ along a trajectory of the corresponding continuous time system.

The term $\sum_{i=0}^{n(x)-1} \eta(F_*^i(x))$ is not multiplied by δ. However, as δ decreases to 0, $\eta(\cdot)$ decreases to 0 faster than δ. Since $n(x)$ increases linearly as δ decreases, it follows that the term vanishes at the limit $\delta \to 0$.

The fourth term is the sum $\sum_{i=0}^{n(x)-1} \frac{\delta}{4}\epsilon(H_i(x))$. The interpretation of this term is similar to that of the second term, only along the trajectory $H_i(x)$, $i = 0, \cdots n(x)-1$. This trajectory is a slight variation of the trajectory $F_*^i(x)$, as can be seen from the definition of $H_i(x)$ (3.30).

The fifth term is the sum of the sums $\frac{\delta}{4}\sum_{j=0}^{m_i(x)} \epsilon(F_{k-i+1}^j(H_i(x)))$, $i = 0, \cdots, n(x) - 1$. Clearly, as the approximation error vanishes uniformly, the term vanishes. Each of these sums approximates the integral of the error along the trajectory that starts off at $H_i(x)$ and ends when the state has become less or equal to $\epsilon(H_i(x))$. As δ decreases towards 0, the term approaches the a finite time horizon integral of the approximation error. It then looks like the overall term might increase as the discretization constant δ decreases, since $n(x)$ then increases linearly with δ. The above discussion suggests that, *when given the option to use more computational power, it is more preferable to use it towards improving the cost function approximation error ϵ by making additional trajectory simulations, rather than towards more accurate simulations by decreasing the constant δ.*

The sixth term is the sum of the sums $\sum_{j=0}^{m_i} \eta(F_{k-i+1}^j(H_i(x)))$. As δ decreases towards 0, the error η in general decreases roughly at a rate δ^2, as it follows from Taylor expansion theory. Since m_i increases roughly at the rate $1/\delta$, it follows that each of these sums decreases roughly as δ. Since $n(x)$ increases roughly at a rate δ, it follows that the overall term might converge towards a constant as δ vanishes. This shows that our chosen strategy of approximate policy update (1.17) or (3.7) may result in an amount of suboptimality that does not vanish even if we choose very small δ. At this point, it is not clear whether this term can be made to collapse to zero with a different line of proof or with a somewhat different set of assumptions. It can be argued that the upper bound collapses to zero if the policy update is implemented exactly, instead of the form (3.7) which results from the Taylor expansion simplification (3.6) (see the remark below). The fact that the sixth term cannot be guaranteed to collapse to 0 as $\delta \rightarrow 0$ might account for the particular way in which the policy update step is implemented.

In conclusion, the upper bound on suboptimality goes to a minimum as the approximation error ϵ vanishes uniformly. The minimum may not be 0, probably as a result of our chosen strategy of approximate policy iteration implementation. Thereby, a tradeoff is revealed between practical implementability of policy iteration and the degree to which optimality can be achieved.

An important remark follows which supports the last argument:

Remark:

We can show that if we do not assume a certain implementation strategy for approximate policy iteration, and we simply assume some cost function approximation error and some appropriate policy update error along the lines of Proposition 6.2 in Chapter 6 of [11], the resulting error bounds on the suboptimality *do vanish as both the approximation error and the update error vanish.*

We compare this result to the similar results by Bertsekas and Tsitsiklis in Propositions 6.2 and 6.3 of [11] for the finite state discounted problem and stochastic shortest path problem, respectively, from which this result draws a lot. The key element of Bertsekas and Tsitsiklis' results is the knowledge of the rate at which cost is accumulated along a trajectory. A similar notion in our derivation is introduced via the assumption that the upper bounds $M(x,\xi)$ and $N(x,\xi)$ exist (cf. (3.27) and (3.28) respectively). As discussed above, the upper bounds of Bertsekas and Tsitsiklis vanish as the approximation and update errors collapse to 0, whereas the bound (3.32) does not, for the reasons discussed above. Finally, in Theorem 3.2.1 the maximum number of iterations necessary for achieving our upper bound is given, whereas in the results of Bertsekas and Tsitsiklis it is only shown that their upper bounds are achieved asymptotically as the number of iterations goes to infinity. It should be kept in mind that that we only deal here with deterministic problems. For deterministic finite state shortest path problems, the results in [11] can also be extended to provide a bound on the number of iterations.

3.3 Proof of the Theorem

We now proceed with deriving the upper bound (3.32).

Proof of Theorem 3.2.1:

Throughout the proof, whenever we write the symbol of a function without an argument, x is implied as the argument (e.g. $T_{\mu_{k+1}}J_{\mu_k}$ in place of $T_{\mu_{k+1}}J_{\mu_k}(x)$). Using the definition (3.16) of ζ_2 and Taylor's expansion, we have:

$$T_{\nu_{k+1}}J_{\mu_k} = \delta x^T x + \delta \nu_{k+1}^2 + J_{\mu_k} + \delta L_f J_{\mu_k} + \delta L_g J_{\mu_k}\nu_{k+1} + \zeta_2 \tag{3.33}$$

By plugging (3.33) and (3.14) into (3.19), we obtain

$$\begin{aligned} TJ_{\mu_k} &= \delta x^T x + \frac{\delta}{4}(L_g J_{\mu_k})^2 + J_{\mu_k} + \delta L_f J_{\mu_k} + \frac{\delta}{2} L_g J_{\mu_k}(-L_g J_{\mu_k}) + \zeta_2 - \xi - \zeta_2 + \zeta_1 \\ &= \delta x^T x - \frac{\delta}{4}(L_g J_{\mu_k})^2 + J_{\mu_k} + \delta L_f J_{\mu_k} - \xi + \zeta_1 \end{aligned} \tag{3.34}$$

Using appropriate Taylor expansions and the form $\mu_{k+1} = -\frac{1}{2}L_g\tilde{J}_{\mu_k}$, we write the difference $T_{\mu_{k+1}}J_{\mu_k} - T_{\mu_{k+1}}\tilde{J}_{\mu_k}$ as

$$T_{\mu_{k+1}}J_{\mu_k} - T_{\mu_{k+1}}\tilde{J}_{\mu_k} = \delta(x^T x + \tfrac{(L_g\tilde{J}_{\mu_k})^2}{4}) + J_{\mu_k} + \delta L_f J_{\mu_k} + \tfrac{\delta}{2}(L_g J_{\mu_k})(-L_g\tilde{J}_{\mu_k}) + \zeta_3 -$$

$$\begin{aligned} & -\delta(x^Tx + \tfrac{(L_g\tilde{J}_{\mu_k})^2}{4}) - \tilde{J}_{\mu_k} - \delta L_f\tilde{J}_{\mu_k} - \tfrac{\delta}{2}(L_g\tilde{J}_{\mu_k})(-L_g\tilde{J}_{\mu_k}) - \tilde{\zeta}_2 \\ = \quad & \delta\{(L_fJ_{\mu_k} - L_f\tilde{J}_{\mu_k}) + \tfrac{1}{2}L_g\tilde{J}_{\mu_k}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})\} \\ & +J_{\mu_k} - \tilde{J}_{\mu_k} + \zeta_3 - \tilde{\zeta}_2. \end{aligned} \tag{3.35}$$

By combining (3.11) and (3.35), we obtain

$$\begin{aligned} T_{\mu_{k+1}}J_{\mu_k} = \; & \delta\{(L_fJ_{\mu_k} - L_f\tilde{J}_{\mu_k}) + \tfrac{1}{2}L_g\tilde{J}_{\mu_k}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})\} \\ & +J_{\mu_k} - \tilde{J}_{\mu_k} + \zeta_3 - \tilde{\zeta}_2 + T\tilde{J}_{\mu_k} + \tilde{\xi} + \tilde{\zeta}_2 - \tilde{\zeta}_1. \end{aligned} \tag{3.36}$$

By plugging (3.9) into (3.11), we obtain

$$\begin{aligned} T\tilde{J}_{\mu_k} = \; & T_{\mu_{k+1}}\tilde{J}_{\mu_k} - \tilde{\xi} - \tilde{\zeta}_2 + \tilde{\zeta}_1 \\ = \; & \delta(x^Tx + \tfrac{(L_g\tilde{J}_{\mu_k})^2}{4}) + \tilde{J}_{\mu_k} + \delta L_f\tilde{J}_{\mu_k} + \tfrac{1}{2}\delta(L_g\tilde{J}_{\mu_k})(-L_g\tilde{J}_{\mu_k}) + \tilde{\zeta}_2 - \tilde{\xi} - \tilde{\zeta}_2 + \tilde{\zeta}_1 \\ = \; & \delta(x^Tx - \tfrac{(L_g\tilde{J}_{\mu_k})^2}{4}) + \tilde{J}_{\mu_k} + \delta L_f\tilde{J}_{\mu_k} - \tilde{\xi} + \tilde{\zeta}_1 \end{aligned} \tag{3.37}$$

We subtract (3.34) from (3.37) and we have:

$$\begin{aligned} T\tilde{J}_{\mu_k} = \; & TJ_{\mu_k} + \delta x^Tx - \tfrac{\delta}{4}(L_g\tilde{J}_{\mu_k})^2 + \tilde{J}_{\mu_k} + \delta L_f\tilde{J}_{\mu_k} + \tilde{\zeta}_1 - \tilde{\xi} - \\ & -\delta x^Tx + \tfrac{\delta}{4}(L_gJ_{\mu_k})^2 - J_{\mu_k} - \delta L_fJ_{\mu_k} + \xi - \zeta_1 \\ = \; & TJ_{\mu_k} + \delta\{\tfrac{(L_gJ_{\mu_k})^2}{4} - \tfrac{(L_g\tilde{J}_{\mu_k})^2}{4} + L_f\tilde{J}_{\mu_k} - L_fJ_{\mu_k}\} + \tilde{J}_{\mu_k} - J_{\mu_k} \\ & +\tilde{\zeta}_1 - \tilde{\xi} + \xi - \zeta_1. \end{aligned} \tag{3.38}$$

By plugging (3.38) for $T\tilde{J}_{\mu_k}$ into (3.36), we have:

$$\begin{aligned} T_{\mu_{k+1}}J_{\mu_k} \; = \; & \delta\{(L_fJ_{\mu_k} - L_f\tilde{J}_{\mu_k}) + \frac{1}{2}L_g\tilde{J}_{\mu_k}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})\} + J_{\mu_k} - \tilde{J}_{\mu_k} + \zeta_3 - \tilde{\zeta}_2 \\ & +\tilde{\xi} + \tilde{\zeta}_2 - \tilde{\zeta}_1 + TJ_{\mu_k} + \delta\{\frac{(L_gJ_{\mu_k})^2}{4} - \frac{(L_g\tilde{J}_{\mu_k})^2}{4} + L_f\tilde{J}_{\mu_k} - L_fJ_{\mu_k}\} \end{aligned}$$

$$
\begin{aligned}
&+\tilde{J}_{\mu_k} - J_{\mu_k} + \tilde{\zeta}_1 - \tilde{\xi} + \xi - \zeta_1 \\
=\;& TJ_{\mu_k} + \frac{\delta}{4}\{(L_g\tilde{J}_{\mu_k})^2 + (L_gJ_{\mu_k})^2 - 2(L_g\tilde{J}_{\mu_k})(L_gJ_{\mu_k})\} + \xi - \zeta_1 + \zeta_1 \qquad (3.39) \\
\leq\;& J_{\mu_k} + \frac{\delta}{4}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})^2 + \eta. \qquad (3.40)
\end{aligned}
$$

We used the fact that $TJ_{\mu_k} \leq J_{\mu_k}$, which follows from definition of TJ_{μ_k}, and the definition (3.20) of the function η. Finally, by using definition (3.21) we write (3.40) as

$$
T_{\mu_{k+1}}J_{\mu_k}(x) \leq J_{\mu_k}(x) + \frac{\delta}{4}\epsilon_k(x) + \eta(x) \qquad (3.41)
$$

It can be easily verified by definition of $T_{\mu_{k+1}}J_{\mu_{k+1}}$ (3.4) that

$$
T_{\mu_{k+1}}J_{\mu_{k+1}} = J_{\mu_{k+1}}. \qquad (3.42)
$$

We subtract (3.42) from (3.41) to obtain

$$
T_{\mu_{k+1}}J_{\mu_k} - T_{\mu_{k+1}}J_{\mu_{k+1}} \leq J_{\mu_k} - J_{\mu_{k+1}} + \frac{\delta}{4}\epsilon_k + \eta
$$

which, after bringing J_{μ_k} and $J_{\mu_{k+1}}$ to the left and using definition (3.4), leads to

$$
\begin{aligned}
J_{\mu_{k+1}} - J_{\mu_k} \leq\;& T_{\mu_{k+1}}J_{\mu_{k+1}} - T_{\mu_{k+1}}J_{\mu_k} + \tfrac{\delta}{4}\epsilon_k + \eta \\
=\;& \delta(x^Tx + \tfrac{(-L_g\tilde{J}_{\mu_k})^2}{2}) + J_{\mu_{k+1}}(F^1_{k+1}(x)) - \delta(x^Tx + \tfrac{(-L_g\tilde{J}_{\mu_k})^2}{4}) \\
& -J_{\mu_k}(F^1_{k+1}(x)) + \tfrac{\delta}{4}\epsilon_k + \eta \\
=\;& J_{\mu_{k+1}}(F^1_{k+1}(x)) - J_{\mu_k}(F^1_{k+1}(x)) + \tfrac{\delta}{4}\epsilon_k + \eta. \qquad (3.43)
\end{aligned}
$$

We now apply (3.43) recursively to the difference $J_{\mu_{k+1}}(F^1_{k+1}(x)) - J_{\mu_k}(F^1_{k+1}(x))$ to obtain

$$
J_{\mu_{k+1}}(x) - J_{\mu_k}(x) \leq J_{\mu_{k+1}}(F^2_{k+1}(x)) - J_{\mu_k}(F^2_{k+1}(x)) +
$$

$$+\tfrac{\delta}{4}\epsilon_k(F^1_{k+1}(x))\eta(F^1_{k+1}(x)) +$$
$$+\tfrac{\delta}{4}\epsilon_k(x) + \eta(x). \tag{3.44}$$

Proceeding recursively up to the time step $M_k(x, \frac{\delta}{4}\epsilon_k(x))$ along the trajectory that starts at x, we obtain

$$\begin{aligned} J_{\mu_{k+1}}(x) - J_{\mu_k}(x) \leq\ & \tfrac{\delta}{4}\epsilon_k(x) + \tfrac{\delta}{4}\textstyle\sum_{i=0}^{M_k(x,\frac{\delta}{4}\epsilon_k(x))} \epsilon_k(F^i_{k+1}(x)) + \\ & + \textstyle\sum_{i=0}^{M_k(x,\frac{\delta}{4}\epsilon_k(x))} \eta(F^i_{k+1}(x)). \end{aligned} \tag{3.45}$$

At this point, we will work towards establishing a similar inequality involving μ^* and J^*. The intermediate result (3.45) will then be used in order to conclude the proof. Observe that $TJ_{\mu_k} \leq T_\mu J_{\mu_k}$ for any policy μ, including μ^*. This is a direct consequence of the definitions (3.3) and (3.4). Observe, also, that $T_{\mu^*}J^* = J^*$. Using these facts, equality (3.39) implies

$$\begin{aligned} T_{\mu_{k+1}}J_{\mu_k} \leq\ & T_{\mu^*}J_{\mu_k} + \tfrac{\delta}{4}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})^2 + \eta \\ =\ & T_{\mu^*}J_{\mu_k} - T_{\mu^*}J^* - J^* + \tfrac{\delta}{4}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})^2 + \eta \\ =\ & \delta\{x^Tx + (\mu^*)^2\} + J_{\mu_k}(F^1_*(x)) - \delta\{x^Tx + (\mu^*)^2\} \\ & -J^*(F^1_*(x)) + J^* + \tfrac{\delta}{4}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})^2 + \eta \\ =\ & J_{\mu_k}(F^1_*(x)) - J^*(F^1_*(x)) + J^* + \tfrac{\delta}{4}(L_g\tilde{J}_{\mu_k} - L_gJ_{\mu_k})^2 + \eta \end{aligned} \tag{3.46}$$

Again, we observe that $T_{\mu_{k+1}}J_{\mu_{k+1}} = J_{\mu_{k+1}}$. Therefore,

$$T_{\mu_{k+1}}J_{\mu_k} = J_{\mu_{k+1}} - T_{\mu_{k+1}}J_{\mu_{k+1}} + T_{\mu_{k+1}}J_{\mu_k}. \tag{3.47}$$

By subtracting (3.47) from (3.46), by using the definition of ϵ_k and (3.45), and by expanding $T_{\mu_{k+1}}J_{\mu_{k+1}}$ and $T_{\mu_{k+1}}J_{\mu_k}$, we obtain

$$\begin{aligned}
J_{\mu_{k+1}}(x) - J^*(x) \leq \; & J_{\mu_k}(F_*^1(x)) - J^*(F_*^1(x)) + \tfrac{\delta}{4}\epsilon_k(x) + \eta(x) + T_{\mu_{k+1}}J_{\mu_{k+1}} - T_{\mu_{k+1}}J_{\mu_k} \\
= \; & J_{\mu_k}(F_*^1(x)) - J^*(F_*^1(x)) + \tfrac{\delta}{4}\epsilon_k(x) + \delta\{x^Tx + \mu_{k+1}(x)^2\} \\
& + J_{\mu_{k+1}}(F_{k+1}^1(x)) - \delta\{x^Tx + \mu_{k+1}(x)^2\} - J_{\mu_k}(F_{k+1}^1(x)) + \eta(x) \\
= \; & J_{\mu_k}(F_*^1(x)) - J^*(F_*^1(x)) + \tfrac{\delta}{4}\epsilon_k(x) + J_{\mu_{k+1}}(F_{k+1}^1(x)) - \\
& - J_{\mu_k}(F_{k+1}^1(x)) + \eta(x) \qquad (3.48)
\end{aligned}$$

Consider $k \geq 1$. By applying the last inequality on the difference $J_{\mu_k}(F_*^1(x)) - J^*(F_*^1(x))$, we obtain

$$\begin{aligned}
J_{\mu_k}(F_*^1(x)) - J^*(F_*^1(x)) \leq \; & J_{\mu_{k-1}}(F_*^2(x)) - J^*(F_*^2(x)) + \tfrac{\delta}{4}\epsilon_{k-1}(F_*^1(x)) \\
& + J_{\mu_k}(F_k^1(F_*^1(x))) - J_{\mu_{k-1}}(F_k^1(F_*^1(x))) + \eta(F_*^1(x)) \quad (3.49)
\end{aligned}$$

By plugging (3.49) into (3.48) we obtain, for $k \geq 1$,

$$\begin{aligned}
J_{\mu_{k+1}}(x) - J^*(x) \leq \; & J_{\mu_{k-1}}(F_*^2(x)) - J^*(F_*^2(x)) + \tfrac{\delta}{4}\textstyle\sum_{i=0}^1 \epsilon_{k-i}(F_*^i(x)) + \sum_{i=0}^1 \eta(F_*^i(x)) + \\
& + \textstyle\sum_{i=0}^1\{J_{\mu_{k+1-i}}(F_{k+1-i}^1(F_*^i(x))) - J_{\mu_{k-i}}(F_{k+1-i}^1(F_*^i(x)))\} \quad (3.50)
\end{aligned}$$

In turn, we may write an inequality similar to (3.49), for $k \geq 2$, for the difference $J_{\mu_{k-1}}(F_*^2(x)) - J^*(F_*^2(x))$, and then replace in (3.50). Then the left hand side of the latter will include the term $J_{\mu_{k-2}}(F_*^3(x)) - J^*(F_*^3(x))$. We repeat the same procedure for this term. By repeating the above steps up to n and by using the definition (3.29) of $n(x)$, we obtain for $k \geq n(x) - 1$:

$$\begin{aligned}
J_{\mu_{k+1}}(x) - J^*(x) \leq \; & J_{\mu_{k+1-n(x)}}(F_*^n(x)) - J^*(F_*^n(x)) \\
& + \tfrac{\delta}{4}\textstyle\sum_{i=0}^{n(x)-1} \epsilon_{k-i}(F_*^i(x)) + \sum_{i=0}^{n(x)-1} \eta(F_*^i(x))
\end{aligned}$$

$$+\sum_{i=0}^{n(x)-1}\{J_{\mu_{k+1-i}}(F^1_{k+1-i}(F^i_*(x))) - J_{\mu_{k-i}}(F^1_{k+1-i}(F^i_*(x)))\} \quad (3.51)$$

We now use inequality (3.45) to replace the terms of the last sum in (3.51). At this point comes handy the notation $H_i(x)$, defined in (3.30):

$$\begin{aligned} J_{\mu_{k+1}}(x) - J^*(x) \leq \quad & J_{\mu_{k+1-n(x)}}(F^n_*(x)) - J^*(F^n_*(x)) \\ & +\tfrac{\delta}{4}\sum_{i=0}^{n(x)-1}\epsilon_{k-i}(F^i_*(x)) + \sum_{i=0}^{n(x)-1}\eta(F^i_*(x)) \\ & +\sum_{i=0}^{n(x)-1}\{\tfrac{\delta}{4}\epsilon_{k-i}(H_i(x)) + \tfrac{\delta}{4}\sum_{j=0}^{m_i}\epsilon_{k-i}(F^j_{k-i+1}(H_i(x))) \\ & +\sum_{j=0}^{m_i}\eta(F^j_{k-i+1}(H_i(x)))\} \end{aligned} \quad (3.52)$$

where we also used the definition (3.31) of m_i, and indirectly, that of $\epsilon(x,\xi)$ (3.26). The latter, in conjuction with the definition of $n(x)$ is used in the final step, which follows:

$$\begin{aligned} J_{\mu_{k+1}}(x) - J^*(x) \leq \quad & \epsilon(x) + \tfrac{\delta}{4}\sum_{i=0}^{n(x)-1}\epsilon(F^i_*(x)) + \sum_{i=0}^{n(x)-1}\eta(F^i_*(x)) + \\ & +\sum_{i=0}^{n(x)-1}\{\tfrac{\delta}{4}\epsilon(H_i(x)) + \tfrac{\delta}{4}\sum_{j=0}^{m_i}\epsilon(F^j_{k-i+1}(H_i(x))) + \\ & +\sum_{j=0}^{m_i}\eta(F^j_{k-i+1}(H_i(x)))\}. \end{aligned} \quad (3.53)$$

Finally, we replace k for $k+1$ in (3.53) and (3.32) follows. ■

Chapter 4

Design of Stabilizing Controllers for Unstable Systems

Instability is the most serious problem facing a control engineer. In practice, a system is unstable when a trajectory of the system starting from a non-equilibrium point grows out of some finite bounds. Exceeding those bounds is unacceptable for a device. The bounds may represent some worst case performance specification limits or, even more importantly, some safety limits. As an example of the latter case, consider the longitudinal dynamics of a conventional aircraft, controlled exclusively by its aerodynamic surfaces and thrust, but not by any novel ways like thrust vectoring. If the angle of attack of such an aircraft ever exceeds the stall angle threshold, it enters a situation from which there is not enough actuator power to recover. Then, the airplane is due to flip over and, eventually crash out of control.

The mathematical models that are typically used to describe the dynamics of a device do not recognize these limitations. They are based on the system's dynamic behavior inside the limits of acceptable operation, without accounting for the state magnitude physical limitations. Instability for these models amounts to their trajectories starting from a non-equilibrium point growing unbounded. Such models might have several advantages. A potential advantage is linearity. There exists a variety of control design tools for linear models. If a system can be modeled by a linear model with a high degree of accuracy inside the region of acceptable operation, then good controllers can be designed via linear control design tools which ensure that the dynamic variables of the system will never exceed the safety limitations. In that case, not including the limitations in the model, makes sure that the design task is relatively straightforward, and the designed controllers are such that the closed loop system does indeed operate desirably, within the specification or safety limits. Even for systems that we model with nonlinear models, we keep the model simpler and less nonlinear by not including these limitations, thus, making the use of analytical approaches for the control design problem easier.

It turns out, however, that this strategy potentially has the exact opposite effect on difficulty as far as computational methods are concerned. In particular, policy iteration fails for models that allow the dynamic variables to grow infinite in size. To see this, consider the optimal control problem addressed in this thesis. Clearly, the cost function $J(x)$ of the type (1.9) for an unstable discrete time system whose state grows unbounded is infinite for all $x \in \mathbf{R}^n$. This makes policy iteration an impossible method. Consider the policy update step (1.16). The updated controller at a state x is derived on the basis of a comparison between the cost functions among the states that are neighboring x. If the value of the cost is ∞, such a comparison is meaningless. This fact is recognized in the (exact) dynamic programming theory. All convergence results assume knowledge of a certain initial policy under which the cost function is finite for all states. The reader is referred to Chapter 1 and Section 2.2.2 of [10].

In this chapter, we show that by incorporating bounds on the dynamic variables in models of unstable systems, approximate policy iteration becomes a feasible method for designing stable controllers. That is, we pose an optimal control problem for the new model of the system, the solution of which is a stable controller, as will be shown. The suboptimal controller that we obtain with approximate policy iteration should be stable as well, provided that the approximation errors are small. The resulting stable controller forces any trajectory that starts within the bounds to converge to the equilibrium point or just remain within the bounds. As argued above, this approach makes very good sense from an application point of view, as well.

Control over a *selected* bounded region around the origin is known as semiglobal control in the literature. Recently, several researchers have proposed semiglobal controllers for different classes of nonlinear systems [40, 38, 69, 20]. The dual advantage of practicality and feasibility of computational methods has been exploited in [39]. More applications of semiglobal control can be found in [77, 19].

4.1 Model Formulation

We are given a model

$$x_{t+1} = f(x_t) + G(x_t)u, \qquad x_0 \in \mathbf{R}^m, u \in U \subset \mathbf{R}^m \tag{4.1}$$

of a nonlinear system in discrete time, where U is a bounded region that includes 0. We assume that there exists no known controller which stabilizes the above model. The discrete time model is derived from a continuous time model of the form (1.1). Notice, that the discrete time representation (4.1) is more general than and includes representation (1.7) which has been used throughout this work. The requirement to specify a bounded region U that u belongs to, is needed for the theoretical guarantees of the approach to be developed in the sequel of this chapter. We are free to select U as large as we desire, but bounded. This is by all means a realistic assumption from a practical point of view, too, as in all real life applications such bounds exist. The origin 0 is an unstable equilibrium point of the open loop system. The trajectories of (4.1) with $u = 0$ starting from almost any nonzero state x_0 at time 0 grow unbounded:

$$\lim_{t\to\infty} \|x_t\| = \infty, \qquad \text{if} \quad x_0 \neq 0 \tag{4.2}$$

The *training region* TR, which includes the origin, is specified by the control engineer. It is the region for which a stable controller is desirable; that is, a controller $\mu(x)$ such that any trajectory of the closed loop system

$$x_{t+1} = f(x_t) + G(x_t)\mu(x_t), \qquad x_0 \in \mathrm{TR} \tag{4.3}$$

starting off from a point x_0 in TR will remain bounded, or converge to the origin. The control engineer also specifies the *region of acceptable operation* RAO, such that TR $\subset$ RAO. Both regions TR and RAO are compact, that is, closed and bounded. For example, they may be of the form

$$\begin{aligned} \mathrm{TR} &= \{x : \|x\| \leq R_{TR}\} \\ \mathrm{RAO} &= \{x : \|x\| \leq R_{RAO}\} \\ 0 &< R_{TR} \leq R_{RAO} \end{aligned} \tag{4.4}$$

In that case, the surface $\{x : \|x\| = R_{TR}\}$ ($\{x : \|x\| = R_{RAO}\}$) is the boundary of region TR (region RAO). See Figure 4.1 The following model is then adopted to describe the underlying physical process:

$$x_{t+1} = \begin{cases} f(x_t) + G(x_t)u & \text{if } x_t \in RAO \\ x_t & \text{otherwise} \end{cases} \tag{4.5}$$

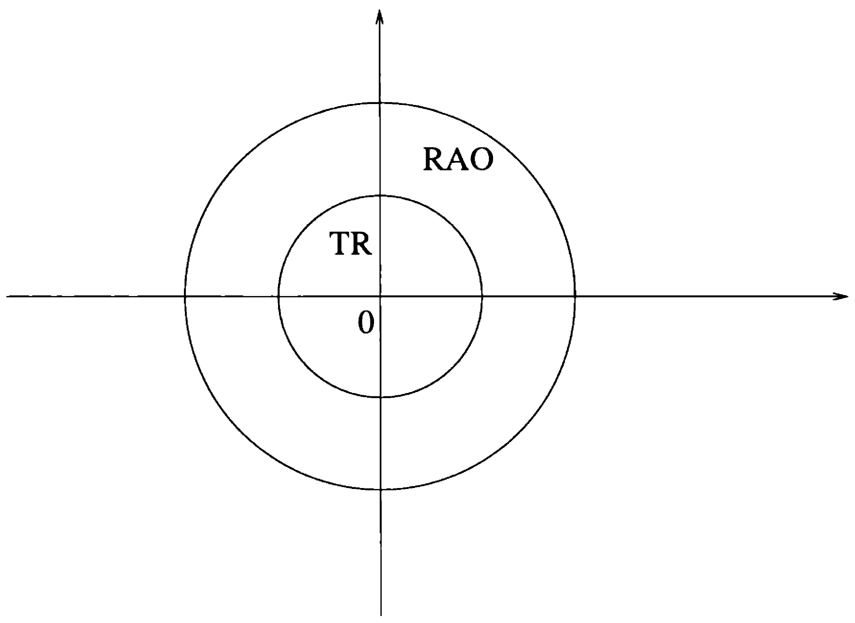

Figure 4.1: The regions RAO and TR

Let $u = \mu(x)$ be any controller (policy). Consider a state $x_0 \in$ TR. Then, we denote by $x^{\mu}_{x_0}(t)$ the trajectory of (4.5) controlled by μ which starts off at x_0 at time $t = 0$:

$$\begin{aligned} x^{\mu}_{x_0}(t+1) &= \begin{cases} f(x^{\mu}_{x_0}(t)) + G(x^{\mu}_{x_0}(t))\mu(x^{\mu}_{x_0}(t)), & \text{if } x^{\mu}_{x_0}(t) \in RAO \\ x^{\mu}_{x_0}(t) & \text{otherwise} \end{cases} \\ x^{\mu}_{x_0}(0) &= x_0. \end{aligned} \tag{4.6}$$

Notice that we changed the notation a little bit in (4.6), in order to avoid subscript jamming: the dependence on the discrete time index appears in a parenthesis following the state vector, rather than subscripted as in (4.1).

Definition 4.1.1 *Given a controller (policy) μ and a value of the state $x_0 \in TR$, the instant of destabilization $N_\mu(x_0)$ with respect to RAO is the instant at which the trajectory $x^{\mu}_{x_0}$ leaves RAO. For example, if RAO is of the type $RAO = \{x : \|x\| \leq R_{RAO}\}$, then*

$$N_\mu(x_0) = \min\{t \in \mathbf{Z}^+ : \|x^{\mu}_{x_0}(t)\| > R_{RAO}\} \tag{4.7}$$

Optimal Control Problem: We are stating an optimal control problem for (4.5). The optimization objective is finding a policy μ^* such that

$$\begin{aligned}\mu^* = \arg\min_{\mu} \{ \quad & \textstyle\sum_{t=0}^{N_\mu(x_0)} \alpha^t \left(x_{x_0}^\mu(t)^T Q x_{x_0}^\mu(t) + \mu(x_{x_0}^\mu(t))^T R \mu(x_{x_0}^\mu(t))\right) + \\ & + M \cdot \textstyle\sum_{t=N_\mu(x_0)+1}^{\infty} \alpha^t \left(x_{x_0}^\mu(t)^T Q x_{x_0}^\mu(t) + \mu(x_{x_0}^\mu(t))^T R \mu(x_{x_0}^\mu(t))\right)\Big\}, \\ & \text{for every } x_0 \in \text{TR},\end{aligned} \tag{4.8}$$

where $\alpha \in (0,1)$ is a cost *discount factor*, $M \geq 1$ is the *unacceptable operation weight factor*, Q and R are symmetric, positive definite state and control weight factors respectively.

Remark: Selecting M to be large, is motivated by our desire to heavily penalize the unstable part of a trajectory so as to make the generated policies pay a high price for going out of the bounds imposed by RAO. As expected, it turns out to be an instrumental element of the stability proofs that follow in the next section (part II of Proposition 4.2.1).

We denote the cost function associated with a certain policy μ, with discount factor α and unacceptable operation weight factor M, by $J_\mu^{(\alpha,M)}$:

$$\begin{aligned}J_\mu^{(\alpha,M)}(x_0) \triangleq \{ \quad & \textstyle\sum_{t=0}^{N_\mu(x_0)} \alpha^t \left(x_{x_0}^\mu(t)^T Q x_{x_0}^\mu(t) + \mu(x_{x_0}^\mu(t))^T R \mu(x_{x_0}^\mu(t))\right) + \\ & + M \cdot \textstyle\sum_{t=N_\mu(x_0)+1}^{\infty} \alpha^t \left(x_{x_0}^\mu(t)^T Q x_{x_0}^\mu(t) + \mu(x_{x_0}^\mu(t))^T R \mu(x_{x_0}^\mu(t))\right)\Big\}.\end{aligned} \tag{4.9}$$

The cost function $J_\mu^{(\alpha,M)}$ turns out to be finite for every policy μ:

Proposition 4.1.1 *The cost function $J_\mu^{(\alpha,M)} : TR \rightarrow \mathbf{R}$ is finite, for trajectories $x_{x_0}^\mu(t)$ of the type (4.6).*

Proof:

Since RAO is a bounded region, it follows that $||x^{\mu}_{x_0}(t)||$ is upper bounded by a bound x_{max} , for all policy μ and initial conditions x_0. Similarly, since U is a bounded region all admissible controls $\mu(x)$ for all x are upper bounded by a bound u_{max}. Also, $\alpha \in (0,1)$ and $M \geq 1$ by choice. Therefore, for any x_0 and μ, it follows from (4.9):

$$\begin{aligned} J^{(\alpha,M)}_{\mu}(x_0) &\leq M \cdot \sum_{t=0}^{\infty} \alpha^t \left(x^{\mu}_{x_0}(t)^T Q x^{\mu}_{x_0}(t) + \mu(x^{\mu}_{x_0}(t))^T R \mu(x^{\mu}_{x_0}(t)) \right) \\ &\leq M \cdot \sum_{t=0}^{\infty} \alpha^t \left(\lambda_{max}(Q) ||x^{\mu}_{x_0}(t)||^2 + \lambda_{max}(R) ||\mu(x^{\mu}_{x_0}(t))||^2 \right) \\ &\leq M(\lambda_{max}(Q) x^2_{max} + \lambda_{max}(R) u^2_{max}) \sum_{t=0}^{\infty} \alpha^t \\ &\leq M(\lambda_{max}(Q) x^2_{max} + \lambda_{max}(R) u^2_{max}) \frac{1}{1-\alpha}, \end{aligned} \tag{4.10}$$

where $\lambda_{max}(Q)$ ($\lambda_{max}(R)$) is the maximum eigenvalue of the positive definite matrix Q (R). From (4.10) it follows that $J^{(\alpha,M)}_{\mu}$ is bounded.

■

Finiteness of the cost function for any controller/policy μ is the motivation behind modeling the unstable system in the form (4.5). It allows the use of policy iteration to solve the optimal control problem (4.8). However, at this point it is important to notice that the optimal controller μ^* may not be stable, in the sense of forcing the state to converge to the origin or simply remain close to it! We demonstrate this with the following example:

Example:

Consider a dynamic system

$$x_{t+1} = \begin{cases} 2x_t + u & \text{if } |x_t| \leq 2 \\ x_t & \text{otherwise} \end{cases}, \quad x, u \in \mathbf{R} \tag{4.11}$$

with training region $|x| \leq 2$. We consider an optimal control problem of the type (4.8) with $M = 1$, $\alpha \in (0,1)$, $Q = 1$ and $R = 1$. It is easy to realize that a policy μ cannot be stable unless it satisfies:

$$\mu(x) = \begin{cases} \leq -2x_t & \text{if } x_t \in [0,2] \\ \geq 2x_t & \text{if } x_t \in [-2,0) \end{cases} \tag{4.12}$$

Let μ_{st} be a stable policy. Consider the cost function:

$$\begin{aligned} J^{(\alpha,1)}_{\mu_{st}}(x_0) &= \sum_{t=0}^{\infty} \alpha^t \left(x^{\mu_{st}}_{x_0}(t)^2 + \mu_{st}(x^{\mu_{st}}_{x_0}(t))^2 \right) \\ &\geq x_0^2 + \mu_{st}(x_0)^2 \\ &\geq x_0^2 + 4x_0^2 \\ &= 5x_0^2, \end{aligned} \tag{4.13}$$

for any discount factor $\alpha \in (0,1)$. In contrast, the policy $\mu_n(x) = 0$ for all $x \in [-2,2]$, is clearly not stable. Consider the cost function

$$\begin{aligned} J^{(\alpha,1)}_{\mu_n}(x_0) &= \sum_{t=0}^{\infty} \alpha^t \left(x^{\mu_n}_{x_0}(t)^2 + \mu_n(x^{\mu_n}_{x_0}(t))^2 \right) \\ &\leq \sum_{t=0}^{\infty} \alpha^t \cdot 2^2 \\ &= \frac{4}{1-\alpha} \end{aligned} \tag{4.14}$$

If $\alpha < 0.2$ and for all $|x_0| \geq 1$, then $J^{(\alpha,1)}_{\mu_n}(x_0) < 5 \leq J^{(\alpha,1)}_{\mu_{st}}(x_0)$. Thus, μ_n induces a lower cost than μ_{st} for $|x_0| \geq 1$. It follows that a stable policy μ_{st} cannot be optimal for $\alpha < 0.2$. Therefore,

In general, the solution of the optimal control problem (4.5), (4.8) need not be a stable policy.

The reason behind this is the discount factor α. If there was no discount in the cost, then the optimal policy would be stable. At the same time, the role of the discount factor is instrumental for the purpose of using the policy iteration algorithm for unstable systems.

In the next section, we argue that for a given problem, there are ranges of α (close to one, but smaller) and M (large enough), assuring that the optimal policy for the problem (4.5), (4.8) *is indeed a stable policy.*

4.2 Discounted Problem and Stability of the Optimal Policy

We define a notion of stability and a notion of exponential stability for a system of the type (4.5), with the following definitions:

Definition 4.2.1 *A policy $\mu_s : RAO \to \mathbf{R}$ is called an* **RAO-safe** *policy if, for any $x_0 \in TR$, the trajectory $x_{x_0}^{\mu_s}(t)$ belongs to RAO for every $t \geq 0$.*

The following definition is another version of 2.1.1, and plays a major role in the stability result which follows in the sequel of this section.

Definition 4.2.2 *A RAO-safe policy $\mu_{es} : RAO \to \mathbf{R}$ for system (4.5) is called an* **exponentially stable** *policy if there exist a positive C and a $q \in (0,1)$ such that, for all $x_0 \in TR$,*

$$x_{x_0}^{\mu_{es}}(t)^T Q x_{x_0}^{\mu_{es}}(t) + \mu_{es}(x_{x_0}^{\mu_{es}}(t))^T R \mu_{es}(x_{x_0}^{\mu_{es}}(t)) \leq C x_0^T Q x_0 q^t, \quad \textit{for all } x_0 \in TR. \tag{4.15}$$

We need the definition of a Q-closed ball of radius ϵ in order to subsequently define the notion of an ϵ-attractive policy.

Definition 4.2.3 *The Q-closed ball B_ϵ with center at the origin of $\mathbf{R}^n$ and radius $\epsilon > 0$ is defined to be the set of all $x \in \mathbf{R}^n$ such that $x^T Q x \leq \epsilon$.*

Definition 4.2.4 *A RAO-safe policy μ_{ba} is called ϵ-attractive if every trajectory $x_{x_0}^{\mu_{ba}}(t)$ which starts off at $x_0 \in TR$ goes inside the Q-closed ball B_ϵ, that is, for every $x_0 \in TR$ there exists an instant t_{x_0} such that $x_{x_0}^{\mu_{ba}}(t_{x_0}) \in B_\epsilon$.*

We now give the definition of stabilizability for (4.5):

Definition 4.2.5 *The system (4.5) is* **stabilizable** *if there exists a RAO-safe policy μ_s. The system (4.5) is called* **exponentially stabilizable** *if there exists an exponentially stable policy μ_{es}.*

Determining the stabilizability properties of a nonlinear dynamic system is an issue which has received a great amount of research effort. There exist systematic tests for stabilizability of several classes of nonlinear dynamic systems [33, 1, 66] as well as several contributions studying the general problem of nonlinear stabilizability [44, 32]. In this work, we assume that stabilizability can somehow be examined, and if a system is stabilizable, then policy iteration is the tool we propose for finding a stabilizing policy.

Let us now examine whether the optimal policy of the problem (4.5), (4.8) is stable, or ϵ-attractive and how this depends on the values of α and M.

Proposition 4.2.1 *I. Assume that a system of the form (4.5) is stabilizable. Then, for any given discount factor α, there exists a real $M(\alpha)$ such that the solution of the optimal control problem (4.8) is stable for discount factor α and unacceptable operation weight factor $M \geq M(\alpha)$. The dependence of $M(\alpha)$ on α is*

$$M(\alpha) = \frac{b}{\alpha^2}, \tag{4.16}$$

for some constant $b > 1$.

II. *Assume that (4.5) is exponentially stabilizable and let μ_{es} be an exponentially stabilizing policy. Then, for every $\epsilon > 0$, there exists a number $\alpha_\epsilon \in (0,1)$ such that the solution of the optimal control problem (4.8) is an ϵ-attractive policy for any discount factor $\alpha \in [\alpha_\epsilon, 1)$ and any unacceptable operation weight factor $M \geq 1$.*

Remark: The nonlinear system can be linearized around the origin and an asymptotically stable linear controller μ_{LTI} can be designed for the linearized system. It is well known [76] that the nonlinear closed loop system under that linear controller is then locally asymptotically stable, that is, there exists a neighborhood of the origin such that all trajectories starting off in that neighborhood converge to the origin. The number ϵ can be small enough such that B_ϵ is subset of that neighborhood. Assume that a nonlinear policy μ_{ba} forces all trajectories of the closed loop system under μ_{ba} to go inside B_ϵ. Then, a switching feedback strategy can be implemented such that the controller switches from μ_{ba} to μ_{LTI} as soon as a trajectory reaches B_ϵ. The resulting controlled system is asymptotically stable. This paradigm motivates part II of Proposition 4.2.1.

Proof of Proposition 4.2.1:

I. Consider the set

$$X_f = \{x_f \notin RAO : x_f = f(x) + G(x)u, \quad x \in RAO, u \in U\}. \tag{4.17}$$

This is the set of all possible terminal points of all trajectories which start off inside TR and leave RAO, under all admissible policies μ that are not RAO-safe. We define a number c as

$$c \triangleq \inf_{x_f \in X_f} \{x_f^T Q x_f\} \tag{4.18}$$

that is, c is a lower bound on the value that the function x^TQx takes in X_f. By assumption, there exists a RAO-safe policy μ_s. The cost function $J^\alpha_{\mu_s}(x_0)$ is bounded over $x_0 \in$ RAO (the unacceptable operation weight factor in the superscript is omitted because it is of no meaning for an RAO-safe policy). Let $\bar{J}^\alpha_{\mu_s}$ be an upper bound over $x_0 \in$ RAO. Consider a policy μ_u which is not RAO-safe and some discount factor $\alpha \in (0,1)$. Let the set S_{μ_u} be defined as

$$\begin{aligned} S_{\mu_u} \triangleq \{ \quad & x \in RAO : \exists x_0 \in TR \quad \text{and} \quad t \geq 0 \\ & \text{such that } \; x^{\mu_u}_{x_0}(t) = x, \text{and} \quad N_{\mu_u}(x) = 1\} \end{aligned} \tag{4.19}$$

Consider a point $x_s \in S_{\mu_u}$. Then, we can find a number $M(\alpha) \geq 1$ such that:

$$\begin{aligned} M(\alpha) \sum_{t=N_{\mu_u}(x_s)+1}^{\infty} \alpha^t c = M(\alpha)\sum_{t=2}^{\infty}\alpha^t c = M(\alpha)c\frac{\alpha^2}{1-\alpha} \geq \bar{J}^\alpha_{\mu_s} \\ \geq \sum_{t=0}^{\infty} \alpha^t \left(x^{\mu_s}_{x_s}(t)^T Q x^{\mu_s}_{x_s}(t) + \mu_s(x^{\mu_s}_{x_s}(t))^T R \mu_s(x^{\mu_s}_{x_s}(t))\right) \end{aligned} \tag{4.20}$$

From the definition of c (4.18), it follows that:

$$\begin{aligned} M(\alpha)\sum_{t=2}^{\infty}\alpha^t c \;&\leq\; M(\alpha)\sum_{t=2}^{\infty}\alpha^t \left(x^{\mu_u}_{x_s}(t)^T Q x^{\mu_u}_{x_s}(t)\right) \\ &\leq\; M(\alpha)\sum_{t=2}^{\infty}\alpha^t \left(x^{\mu_u}_{x_s}(t)^T Q x^{\mu_u}_{x_s}(t) + \mu_u(x^{\mu_s}_{x_s}(t))^T R \mu_u(x^{\mu_u}_{x_s}(t))\right) \\ &\leq\; \sum_{t=0}^{1}\alpha^t \left(x^{\mu_u}_{x_s}(t)^T Q x^{\mu_u}_{x_s}(t) + \mu_u(x^{\mu_u}_{x_s}(t))^T R \mu_u(x^{\mu_u}_{x_s}(t))\right) \\ &\quad +M(\alpha)\sum_{t=2}^{\infty}\alpha^t \left(x^{\mu_u}_{x_s}(t)^T Q x^{\mu_u}_{x_s}(t) + \mu_u(x^{\mu_u}_{x_s}(t))^T R \mu_u(x^{\mu_u}_{x_s}(t))\right) \\ &=\; J^{(\alpha,M(\alpha))}_{\mu_u}(x_s) \end{aligned} \tag{4.21}$$

By combining (4.20) with (4.21), we obtain for $x_s \in S_{\mu_u}$:

$$\sum_{t=0}^{\infty} \alpha^t \left(x_{x_s}^{\mu_s}(t)^T Q x_{x_s}^{\mu_s}(t) + \mu_s(x_{x_s}^{\mu_s}(t))^T R \mu_s(x_{x_s}^{\mu_s}(t)) \right) \leq J_{\mu_u}^{(\alpha, M(\alpha))}(x_s). \qquad (4.22)$$

This shows that no RAO-unsafe policy is optimal for $x_s \in S_{\mu_u}$ with respect to (4.8) with discount factor α and weight factor $M(\alpha)$. Considering any other $x_0 \in$ TR, the trajectory $x_{x_0}^{\mu_u}$ has to go through S_{μ_u}. Thus, μ_u cannot be optimal. Therefore, the optimal policy has to be RAO-safe.

The same conclusion follows easily for all weight factors $M > M(\alpha)$, since

$$J_{\mu_u}^{(\alpha, M(\alpha))}(x_0) \leq J_{\mu_u}^{(\alpha, M)}(x_0)$$

for all $M \geq M(\alpha)$, by inspection. It then follows from (4.20) that $M(\alpha)$ can be chosen as:

$$M(\alpha) = \frac{1-\alpha}{c\alpha^2} \bar{J}_{\mu_s}^{\alpha} \qquad (4.23)$$

In order to show (4.16), we define

$$d \triangleq \max_{x \in RAO} x^T Q x + \max_{u \in U} u^T R u. \qquad (4.24)$$

Notice that d is a well defined number, since both RAO and U are compact sets, while $x^T Q x$ and $u^T R u$ are continuous functions. From the definitions of d, $\bar{J}_{\mu_s}^{\alpha}$ it follows that

$$\bar{J}_{\mu_s}^{\alpha} \leq \frac{1}{1-\alpha} d \qquad (4.25)$$

Therefore, it follows from (4.20) that $M(\alpha)$ can be selected as

$$M(\alpha) = \frac{b}{\alpha^2}, \qquad (4.26)$$

where $b = d/c$. It can be easily verified that $d > c$, thus $b > 1$.

II. By assumption, the exponentially stable policy μ_{es} is such that

$$x_{x_0}^{\mu_{es}}(t)^T Q x_{x_0}^{\mu_{es}}(t) + \mu_{es}(x_{x_0}^{\mu_{es}}(t))^T R \mu_{es}(x_{x_0}^{\mu_{es}}(t)) \leq C x_0^T Q x_0 q^t, \quad \text{for all } x_0 \in TR. \tag{4.27}$$

Let D be an upper bound on $x_0^T Q x_0$ over TR, that is, D is such that

$$D \geq x_0^T Q x_0, \qquad \text{for all} \qquad x_0 \in \text{TR} \tag{4.28}$$

Consistent with our notation so far, we denote by $J_{\mu_{es}}^1$ the undiscounted cost function along trajectories $x_{x_0}^{\mu_{es}}$ forced by μ_{es} (where the superscript 1 denotes that $\alpha = 1$). By virtue of (4.27) and (4.28), along any trajectory we have:

$$\begin{aligned} J_{\mu_{es}}^1(x_0) = & \sum_{t=0}^{\infty} \left\{ x_{x_0}^{\mu_{es}}(t)^T Q x_{x_0}^{\mu_{es}}(t) + \mu_{es}(x_{x_0}^{\mu_{es}}(t))^T R \mu_{es}(x_{x_0}^{\mu_{es}}(t)) \right\} \\ \leq & \sum_{t=0}^{\infty} (C x_0^T Q x_0 q^t) \leq \tfrac{1}{1-q} CD \end{aligned} \tag{4.29}$$

which implies that, for every $\eta \in (0, 1)$:

$$\frac{1}{\eta} J_{\mu_{es}}^1(x_0) \leq \frac{1}{\eta(1-q)} CD \tag{4.30}$$

Now, let us assume that there exists a RAO-safe policy μ_s, and some $x_0 \in$ TR such that

$$x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) > \epsilon, \qquad \text{for all} \quad t \geq 0. \tag{4.31}$$

Note that the following arguments also go through if a policy which is not RAO-safe is considered instead of μ_s, provided that the unacceptable operation weight factor M is greater than or equal to 1. Consider such a x_0. Then, there exists a positive integer $N(x_0, \eta)$ such that:

$$\sum_{t=0}^{N(x_0,\eta)} \{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t)\} \geq N(x_0, \eta)\epsilon \geq \frac{CD}{\eta(1-q)} \tag{4.32}$$

Inequality (4.32) shows that $N(x_0, \eta) = \frac{CD}{\eta\epsilon(1-q)}$, which is independent of x_0, therefore we can instead denote the integer by $N(\eta)$. In order to ease notational congestion to some degree, we omit the dependence on η in the sequel and write N instead of $N(\eta)$. By virtue of (4.30) and (4.32) it follows that

$$\begin{aligned}\sum_{t=0}^{N}&\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) + \mu_s(x_{x_0}^{\mu_s}(t))^T R \mu_s(x_{x_0}^{\mu_s}(t))\} \\ &\geq \sum_{t=0}^{N}\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t)\} \geq \frac{CD}{\eta(1-q)} \\ &\geq \frac{1}{\eta} J_{\mu_{es}}^1(x_0)\end{aligned} \tag{4.33}$$

We pick a number $\alpha(\eta) \in (0, 1)$ as

$$\alpha(\eta) = \sqrt[N]{\eta}, \quad \text{where} \quad N = \frac{CD}{\eta\epsilon(1-q)}. \tag{4.34}$$

Again, as with $N(\eta)$, in the sequel we omit the dependence of $\alpha(\eta)$ on η. Then, it follows from (4.34), (4.33) and the definition of $J_{\mu_{es}}^1$ that:

$$\begin{aligned}\sum_{t=0}^{\infty}&\alpha^t\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) + \mu_s(x_{x_0}^{\mu_s}(t))^T R \mu_s(x_{x_0}^{\mu_s}(t))\} \\ &\geq \sum_{t=0}^{N}\alpha^t\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) + \mu_s(x_{x_0}^{\mu_s}(t))^T R \mu_s(x_{x_0}^{\mu_s}(t))\} \\ &\geq \alpha^N \sum_{t=0}^{N}\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) + \mu_s(x_{x_0}^{\mu_s}(t))^T R \mu_s(x_{x_0}^{\mu_s}(t))\} \\ &= \eta \sum_{t=0}^{N}\{x_{x_0}^{\mu_s}(t)^T Q x_{x_0}^{\mu_s}(t) + \mu_s(x_{x_0}^{\mu_s}(t))^T R \mu_s(x_{x_0}^{\mu_s}(t))\} \geq \eta\frac{1}{\eta} J_{\mu_{es}}^1(x_0)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{t=0}^{\infty}\{x_{x_0}^{\mu_{es}}(t)^T Q x_{x_0}^{\mu_{es}}(t) + \mu_{es}(x_{x_0}^{\mu_{es}}(t))^T R \mu_{es}(x_{x_0}^{\mu_{es}}(t))\} \\
&\geq \sum_{t=0}^{\infty}\alpha^t\{x_{x_0}^{\mu_{es}}(t)^T Q x_{x_0}^{\mu_{es}}(t) + \mu_{es}(x_{x_0}^{\mu_{es}}(t))^T R \mu_{es}(x_{x_0}^{\mu_{es}}(t))\} \quad (4.35)
\end{aligned}
$$

The conclusion from (4.35) is that, for every $x_0 \in S_{\mu_s}$, μ_{es} is of lower cost than μ_s, for the optimal control problem with $\alpha = \alpha(\eta)$ (or $\alpha \in [\alpha(\eta), 1)$, as can be easily seen). Therefore, the optimal policy is an ϵ-attractive policy for α given by (4.34). ■

4.3 Approximate Policy Iteration

In the previous section, we concluded that if (4.5) is stabilizable, then we can formulate an optimal control problem of the form (4.8) with the right values of discount factor and unacceptable operation weight factor, whose solution is stable.

To solve the optimal control problem, we use approximate policy iteration. Given a RAO-unsafe policy μ, we select α and M, and perform a number of simulations starting from different initial conditions. By integrating (4.8) along each simulated trajectory, we obtain a sample of the cost function $J_\mu^{(\alpha,M)}$. After selecting an approximation architecture of the type (1.12), we "tune" the architecture weights by means of a least square fit (1.13). The resulting approximate cost function is denoted by $\tilde{J}_\mu$. The approximate policy iteration algorithm then calls for determining the updated policy as

$$
\mu_{min}(x) = \arg\min_{u\in U}\left\{x^T Q x + u^T R u + \alpha \tilde{J}_\mu(f(x) + G(x)u)\right\}, \qquad \text{for all} \qquad x \in RAO \quad (4.36)
$$

Updating the policy in this manner calls for the solution of an optimization problem in u for every x, which is in general hard. Therefore, in practice we simplify (4.36) in some way, e.g. (1.16), and bring it to a form from which we can derive the updated policy in closed form or in a computationally feasible manner. Actually, for the (4.5), (4.8) type of problem we use something similar to (1.16), but we postpone the presentation until Section 4.4. Nevertheless, the policy $\bar{\mu}(x)$ that we obtain does not solve (4.36) exactly. Therefore, two sources of approximation errors are introduced in the process. Let us assume that they are both bounded for $x \in RAO$. The errors and their bounds are expressed as

$$\max_{x \in RAO} |\tilde{J}_\mu - J_\mu^{(\alpha,M)}| \le \epsilon, \tag{4.37}$$

$$\max_{x \in RAO} \Big| \Big\{ x^T Q x + \mu_{min}(x)^T R \mu_{min}(x) + \alpha \tilde{J}_\mu(f(x) + G(x)\mu_{min}(x)) \Big\} - \\ - \Big\{ x^T Q x + \bar{\mu}(x)^T R \bar{\mu}(x) + \alpha \tilde{J}_\mu(f(x) + G(x)\bar{\mu}(x)) \Big\} \Big| \le \delta. \tag{4.38}$$

Assume that starting from an initial policy μ_0 we generate a sequence of policies $\mu_0, \mu_1, \cdots$ in the as described above via approximate policy iteration. Assume, also, that in every iteration the same error bounds ϵ and δ are achieved. Then, it has been shown by Bertsekas and Tsitsiklis and reported in [10], p.42 that

Proposition 4.3.1 *The sequence of policies μ_k generated by the approximate policy iteration algorithm satisfies*

$$\limsup_{k \to \infty} \max_{x \in RAO} (J_{\mu_k}^{(\alpha,M)}(x) - J_*^{(\alpha,M)}(x)) \le \frac{\delta + 2\alpha\epsilon}{(1-a)^2}, \tag{4.39}$$

where $J_*^{(\alpha,M)}$ is the optimal cost function corresponding to the stable optimal policy μ_*. It follows easily that

Corollary 4.3.1 *If the policy iteration algorithm is performed exactly, that is, $\epsilon = 0$*

and $\delta = 0$, then

$$\limsup_{k\to\infty} \max_{x\in RAO} |J_{\mu_k}^{(\alpha,M)}(x) - J_*^{(\alpha,M)}(x)| = 0 \tag{4.40}$$

For nonzero errors ϵ and δ, the asymptotic bound on the difference in performance between μ_k and μ_* is proportional to ϵ and δ.

Remark: The bound $\frac{\delta+2\alpha\epsilon}{(1-a)^2}$ is quite conservative. For example, if we select architectures which ensure that $\mu_k(0) = 0$ for every k, then $J_{\mu_k}^{(\alpha,M)}(0) - J_*^{(\alpha,M)}(0) = 0$ for every k.

On the basis of Corollary 4.3.1, the optimal policy is achievable provided that we have the ability of flawless function approximation and exactly solving the minimization problems (4.36), neither of which is realistic in practice. However, there is still hope. Even in the realistic case of imperfect approximations and minimizations, the policies that are generated are worse in performance than optimal by an amount that only increases linearly with ϵ and δ. Engineering intuition dictates that there are some thresholds ϵ_d and δ_d such that if our approximations do not violate, then the asyptotically resulting suboptimal policies will still be stable. Intuition is favored by the proofs of Proposition 4.2.1. In those, the argument was that *some* stable policy, in general suboptimal, is of lower cost than the unstable ones for the right values of α and M. It follows that:

Corollary 4.3.2 (of Proposition 4.2.1 and Proposition 4.3.1):

In general, the policies μ_s and μ_{es} of the proofs of parts I and II of Proposition 4.2.1 are suboptimal. This implies that indeed, there exist thresholds ϵ_d and δ_d of approximate policy iteration errors, such that if not violated by the approximation errors, the resulting policies as k increases are stable.

4.4 Implementation of Approximate Policy Iteration

We consider the case where we are given an unstable nonlinear dynamic system in continuous time,

$$\dot{x} = f_c(x) + g_c(x)u, \tag{4.41}$$

where u is scalar (for simplicity), $f_c(x)$ and $g_c(x)$ are continuously differentiable. We do not know any stabilizing controller for (4.41), so we select some continuously differentiable $\mu_c(x)$ to close the loop. We define TR and RAO. For a small discretization interval δ, we use the discrete time model

$$x_{t+1} = \begin{cases} x_t + \delta(f_c(x_t) + g_c(x_t)\mu_c(x_t)) & \text{if } x_t \in RAO \\ x_t & \text{otherwise} \end{cases} \tag{4.42}$$

as a simulator of (4.41). We select α and M and define a discounted optimal control problem

$$J_{\mu_c}^{(\alpha,M)}(x_0) \triangleq \Big\{ \sum_{t=0}^{N_{\mu_c}(x_0)} \alpha^t \cdot \delta \left(x_{x_0}^{\mu_c}(t)^T Q x_{x_0}^{\mu_c}(t) + \mu_c(x_{x_0}^{\mu_c}(t))^T R \mu_c(x_{x_0}^{\mu_c}(t)) \right) + \\ + M \cdot \sum_{t=N_{\mu_c}(x_0)+1}^{\infty} \alpha^t \cdot \delta \left(x_{x_0}^{\mu_c}(t)^T Q x_{x_0}^{\mu_c}(t) + \mu_c(x_{x_0}^{\mu_c}(t))^T R \mu_c(x_{x_0}^{\mu_c}(t)) \right) \Big\}. \tag{4.43}$$

By following the procedure described in Section 4.3, we obtain a (smooth, by choice) approximate cost function $\tilde{J}_{\mu_c}$. Then, the policy update rule (4.36) takes the form

$$\arg\min_{u\in U} \left\{ \delta(x^T Q x + u^T R u) + \alpha \tilde{J}_{\mu_c}(x + \delta(f_c(x) + g_c(x)u)) \right\}, \qquad x \in RAO \tag{4.44}$$

In order to express the updated policy in a closed form, we introduce the first order Taylor approximation

$$\tilde{J}_{\mu_c}(x + \delta(f_c(x) + g_c(x)u)) \simeq \tilde{J}_{\mu_c}(x) + \delta L_{f_c(x)}\tilde{J}_{\mu_c}(x) + \delta L_{g_c(x)}\tilde{J}_{\mu_c}(x)u \tag{4.45}$$

We plug (4.45) into (4.44) and we have:

$$\begin{aligned}
&\arg\min_{u\in U}\left\{\delta(x^TQx + u^TRu) + \alpha\tilde{J}_{\mu_c}(x + \delta(f_c(x) + g_c(x)u))\right\} \simeq \\
&\qquad \arg\min_{u\in U}\left\{\delta(x^TQx + u^TRu) + \alpha\left(\tilde{J}_{\mu_c}(x) + \delta L_{f_c(x)}\tilde{J}_{\mu_c}(x) + \delta L_{g_c(x)}\tilde{J}_{\mu_c}(x)u\right)\right\} = \\
&\qquad\qquad \arg\min_{u\in U}\left\{\delta(u^TRu) + \delta\alpha L_{g_c(x)}\tilde{J}_{\mu_c}(x)u\right\}, \\
&\qquad\qquad\qquad \text{for all} \quad x \in RAO
\end{aligned}$$

Therefore, in a similar way to (1.16), the updated policy takes the form

$$\bar{\mu}(x) = -\frac{\alpha}{2}L_{g_c(x)}\tilde{J}_{\mu_c}(x) \tag{4.46}$$

The policy $\bar{\mu}$ is smooth by virtue of smoothness of $\tilde{J}_{\mu_c}$. The approximate policy iteration procedure can then continue the same way.

Chapter 5

Approximation architecture for a class of nonlinear systems

In Chapter 2 we developed bounds on the allowed approximation errors such that a single policy iteration results in a stable policy. We also developed bounds on the allowed approximation error such that a single iteration results in cost improvement. It turns out that the allowed errors are large. In principle, it suffices to compute a limited amount of cost function samples so that approximate policy iteration produces stable and improved policies. The results of Chapter 2 are confidence results. They alone constitute enough justification for using approximate policy iteration in nonlinear control design problems. Nevertheless, the task of cost function approximation can be very demanding, despite the large allowed errors. In particular, in high dimensional problems, it is reasonable to expect that the amount of cost function samples required to generate an acceptable approximation is large and a large amount of computation is required in order to generate them.

In this chapter, we argue that insight into the dynamic structure of a given problem is a potential partial substitute for computations in the cost function approximation task. That would alleviate part of the computational burden and make approximate policy iteration more popular to control designers. In other words, insight may be used to enhance the *practicality* of the method by making the task of cost function approximation more tractable.

As a paradigm, we work with a special class of nonlinear systems, and propose a certain way in which part of the approximation burden can be relaxed. Hopefully, the paradigm can be extended to other classes of systems, as well. However, this special class of systems is important in its own right, since it encompasses a large number of dynamic systems of great practical interest.

We consider nonlinear systems for which it is known that only a few (in many cases, one or two) physical quantities enter the system dynamics in a nonlinear way. The system can then be modeled in the form

$$\begin{bmatrix} \dot{x_N} \\ \dot{x_L} \end{bmatrix} = \begin{bmatrix} f_N(x_N) \\ f_L(x_N) \end{bmatrix} + \begin{bmatrix} A_N(x_N) \\ A_L(x_N) \end{bmatrix} x_L + \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} u, \tag{5.1}$$

where $x_N \in \mathbf{R}^k$ and $x_L \in \mathbf{R}^{n-k}$ are the components of the state that enter the dynamics in a nonlinear or linear fashion, respectively, and $u \in \mathbf{R}^m$ is the control vector. Let the origin $[0^T,\ 0^T]^T$ be an equilibrium point of (5.1), when $u \equiv 0$. That is, $f_N(0) = 0$, $f_L(0) = 0$. Due to the linear fashion in which a number of states enter the dynamics, linear controllers can be designed for fixed values of the x_N-component of the state. The exact procedure is described below. These controllers can be heuristically interpolated to synthesize a controller, which is linear in x_L and nonlinear in x_N; the dependence on x_N arises from the interpolation. The most popular approach of this type, quite commonly used in practice, is gain scheduling. Gain scheduling often proves successful in practice, especially in cases where the dynamics of x_L are dominant in comparison to the dynamics of x_N. In that case, x_L

varies faster than x_N, and the closed loop system overall behaves quite linearly with respect to x_L. In consequence, we expect that the cost function for a gain scheduled closed loop system roughly depends quadratically on x_L. We give the argument some analytical support by showing that in the limit that the size of the nonlinear part of the dynamics is small, the above conjecture is right.

In the cases where the above argument is correct, control design via approximate policy iteration can benefit from it, as it would suggest that a cost function approximation which is quadratic in x_L for a closed loop system of the above type, is likely to be of high accuracy, provided that the dependence on x_N can be adequately approximated. However, having pre-determined the way that the approximation looks like with respect to x_L leaves the designer with the easier task of approximating in the k-dimensional space of x_N. Policy iteration becomes easier to use. As a direct application, approximate policy iteration could be used for improving on controllers designed via gain scheduling or other interpolation methods. Many real life controllers are designed heuristically based on the linearization and interpolation approach described. Since they are heuristic, they come with no performance guarantees. Approximate policy iteration potentially leads to improved controllers.

5.1 Description and assumptions

Consider a *trim point* of the system, that is, a point $[x_{Nf}^T, \ x_{Lf}^T]^T$ satisfying

$$\begin{bmatrix} f_N(x_{Nf}) \\ f_L(x_{Nf}) \end{bmatrix} + \begin{bmatrix} A_N(x_{Nf}) \\ A_L(x_{Nf}) \end{bmatrix} x_{Lf} + \begin{bmatrix} g_N(x_{Nf}) \\ g_L(x_{Nf}) \end{bmatrix} u_f = 0. \qquad (5.2)$$

A trim point is an equilibrium point, of (5.1). Therefore, the dynamics can be linearized around a trim point, and a linear controller can be designed for it, via one of many standard linear control design techniques available. This is a key observation, upon which gain scheduling and other control design approaches for (5.1) are based.

In gain scheduling, stabilizing linear controllers are designed for the system's linearizations around several trim points corresponding to fixed values of the x_N variables. The latter are called the *scheduling variables.* Additional logic is then used to interpolate between these controllers based on measurements of the scheduling variables. The states which enter the dynamics in a nonlinear fashion are treated as scheduling variables, since each linearization of (5.1) corresponds to a different value of x_N. This approach has been studied extensively by several researchers [78, 57, 58, 53], and has been relied upon heavily in several applications. The closed-loop is theoretically guaranteed to be stable only if the scheduling variables x_N evolve slowly with time [57, 58]. However, in practice the gain-scheduled controllers have proved quite reliable in many real world applications. The gain-scheduled controller as described above is linear in the x_L component of the state. An alternative way for designing gain scheduled controllers with certain stability and performance guarantees has been developed recently in [47, 46]. Finally, a recent different formulation of the gain scheduling problem is that of Linear Parameter Varying (LPV) systems [3, 4, 6, 50, 56].

We assume that a gain scheduled controller

$$\mu(x) = \mu_N(x_N)x_L, \tag{5.3}$$

where $\mu_N(x_N)$ is a $m \times (n-k)$ matrix, has been successfully designed for (5.1), such that $[0^T, \ 0^T]^T$ is an exponentially stable equilibrium point of the closed loop system

$$\begin{bmatrix} \dot{x_N} \\ \dot{x_L} \end{bmatrix} = \begin{bmatrix} f_N(x_N) \\ f_L(x_N) \end{bmatrix} + \begin{bmatrix} A_N(x_N) \\ A_L(x_N) \end{bmatrix} x_L + \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} \mu_N(x_N)x_L, \tag{5.4}$$

for every initial condition $[x_{N0}^T, \ x_{L0}^T]^T$ belonging to a bounded region X_0 that contains the origin.

Assumption 5.1.1 *All trajectories starting inside X_0 belong to a bounded region $X_{inv} \supset X_0$, which is a subset of the following compact region X:*

$$X_{inv} \subset X \triangleq \left\{ \begin{bmatrix} x_N \\ x_L \end{bmatrix} \quad s.t. \quad \|x_{N0}\| \leq r_N \text{ and } \|x_{L0}\| \leq r_L \right\}. \tag{5.5}$$

Definition 5.1.1 *A function $f(x)$ satisfies a Lipschitz condition with constant l in an open region D if*

$$\|f(x) - f(y)\| \leq l\|x - y\|$$

for all x and y that belong in D. ■

We state the following assumption on the system dynamics:

Assumption 5.1.2 *The functions $f_N, f_L, A_N, A_L, g_N, g_L, \mu_N$ of x_N are continuously differentiable with respect to x_N in X_{inv}.*

Note that many gain scheduled controllers used in applications are continuous, but non-differentiable with respect to the scheduled variables x_N on some switching surfaces of dimension $n - 1$. Any such controller can be replaced by a continuously differentiable feedback control function, which can be selected arbitrarily close to the original non-differentiable one, and such that all stability and performance properties of the latter are preserved. An example of such a "smoothing" operation is given in Chapter 6. Therefore, Assumption 5.1.2 is a reasonable assumption to make and simplifies the subsequent analysis significantly.

Some Notation:

Consider a continuously differentiable function $f : \mathbf{R}^m \rightarrow \mathbf{R}^n$, which maps a vector

$x = [x_1, \cdots x_m]^T$ to a vector $f(x) = [f_1(x), \cdots, f_n(x)]^T$. Recall from (2.14) that by $\frac{\partial f}{\partial x}(x)$ we denote the $(n \times m)$-matrix whose ij-th element is given by

$$\left[\frac{\partial f}{\partial x}(x)\right]_{ij} = \frac{\partial f_i}{\partial x_j}(x). \tag{5.6}$$

Consider a continuously differentiable function $A : \mathbf{R}^m \to \mathbf{R}^{n_1 \times n_2}$, which maps a vector $x = [x_1, \cdots x_m]^T$ to a matrix $A(x) = [a_{ij}(x)]$. By $\frac{\partial A}{\partial x_l}(x)$ we denote the $(n_1 \times n_2)$-matrix whose ij-th element is given by

$$\left[\frac{\partial A}{\partial x_l}(x)\right]_{ij} = \frac{\partial a_{ij}}{\partial x_l}(x). \tag{5.7}$$

We now make the following assumption on the part of the input vector which corresponds to the scheduling variables x_N:

Assumption 5.1.3 *The part of the input vector that influences the dynamics of the upper part of the state is 0 at $x_N = 0$, that is, $g_N(0) = 0$*

Assumption 5.1.3 ensures that the dynamics of x_N are small even if $\mu_N(x_N)x_L$ is large. Therefore, even if x_{L0} is large, the dynamics of the scheduling variables are still small. This assumption is justified on the basis of the dynamics of several interesting applications, including control of the longitudinal dynamics of aircraft.

Based on the continuous differentiability Assumption 5.1.2, the following can be inferred about the dynamics:

Lemma 5.1.1 *1. The system (5.4) is Lipschitz in X with a constant l_{NL}.*

2. $A_L(x_N)$, $\mu_N(x_N)$, $(g_L\mu_N)(x_N)$ are Lipschitz in X, with constants l_{A_L}, l_μ, $l_{g\mu}$ respectively.

3. The norms of all the terms in the dynamics of the x_N state component, as well *as the nonlinear term $\|f_L(x_N)\|$, are upper bounded:*

$$\|f_N(x_N)\| \le \Delta_1, \quad \|A_N(x_N)\| \le \Delta_2, \quad \|g_N(x_N)\| \le \Delta_3,$$
$$\|g_L(x_N) - g_L(0)\| \le \Delta_4, \quad \|f_L(x_N)\| \le \Delta_5$$

for all $\begin{bmatrix} x_N \\ x_L \end{bmatrix}$ *in X, where Δ_1, Δ_2, Δ_3, Δ_4, Δ_5 are positive numbers.*

4. The following inequalities hold:

$$\|\frac{\partial f_N(x_N)}{\partial x_N}\| \le \Delta_6, \quad \|\frac{\partial f_L(x_N)}{\partial x_N}\| \le \Delta_7, \quad \|\sum_{i=1}^{k}\left(\frac{\partial A_N(x_N)}{\partial x_{N_i}} y_{N_i}\right)\| \le \Delta_8\|y_N\|,$$
$$\|\sum_{i=1}^{k}\left(\frac{\partial A_L(x_N)}{\partial x_{N_i}} y_{N_i}\right)\| \le \Delta_9\|y_N\|, \quad \|\sum_{i=1}^{k}\left(\frac{\partial g_N(x_N)}{\partial x_{N_i}} y_{N_i}\right)\| \le \Delta_{10}\|y_N\|,$$
$$\|\sum_{i=1}^{k}\left(\frac{\partial g_L(x_N)}{\partial x_{N_i}} y_{N_i}\right)\| \le \Delta_{11}\|y_N\|, \quad \|\sum_{i=1}^{k}\frac{\partial \mu_N(x_N)}{\partial x_{N_i}} y_{N_i}\| \le \Delta_{12}\|y_N\|,$$

for every $\begin{bmatrix} x_N \\ x_L \end{bmatrix}$ *in the compact region X, where the upper bounds Δ_6, Δ_7, Δ_8, Δ_9, Δ_{10}, Δ_{11}, Δ_{12} are positive numbers.*

Lemma 5.1.1 follows from continuous differentiability of the dynamics and compactness of X (establishment of inequality $\|g_N(x_N)\| \le \Delta_3$ also uses Assumption 5.1.3). A detailed proof is omitted.

Suppose that the gain-scheduled controller $\mu_N(x_N)x_L$ interpolates differentiably between the linear controllers $C(x_{N1})x_L, \cdots, C(x_{Nq})x_L$. These are designed around a collection of q trim points specified by the values $x_{N1}, \cdots, x_{Nq}$ of the scheduled variables respectively. The i-th controller $C(x_{Ni})$ is designed such that the linear time invariant closed loop system

$$\dot{\zeta} = A_L(x_{Ni})\zeta + g_L(x_{Ni})C(x_{Ni})\zeta \tag{5.8}$$

is exponentially stable. Suppose that the heuristic interpolation is successful, such that the closed loop system (5.4) is exponentially stable.

Assumption 5.1.4 *The linear time invariant systems*

$$\dot{\zeta} = A_L(x_{N0})\zeta + g_L(x_{N0})\mu_N(x_{N0})\zeta \tag{5.9}$$

are exponentially stable for all x_{N0} such that there exists a $x_{L0} \in \mathbf{R}^{n-k}$ for which $[x_{N0}^T,\ x_{L0}^T]^T$ belongs to X_{inv}.

Assumption 5.1.4 essentially views the dynamics of x_L as a stable linear time varying system. It is reasonable to assume that the dynamics of any fixed instance of a linear time varying system is stable. This is precisely the paradigm used for justification of gain scheduled control.

Finally, we state an assumption on the exponential rate of decay of the above systems:

Assumption 5.1.5 *Consider a trajectory $[x_N^T(t),\ x_L^T(t)]^T$ of (5.4) with initial conditions $[x_{N0}^T,\ x_{L0}^T]^T \in X_{inv}$, and a trajectory $\zeta(t)$ of any system of the form (5.9) for x_{N0} such that there exists a x_{L0} for which $[x_{N0}^T,\ x_{L0}^T]^T$ belongs to X_{inv}. It is assumed that $x_L(t)$ as well as $\zeta(t)$ decay exponentially as*

$$||x_L(t)|| \leq \beta ||x_{L0}|| e^{-\gamma t}, \qquad ||\zeta(t)|| \leq \beta ||\zeta_0|| e^{-\gamma t}, \tag{5.10}$$

for all initial conditions $[x_{N0}^T,\ x_{L0}^T]^T \in X_{inv}$ and $\zeta_0 \in \mathbf{R}^{n-k}$.

Remark:

Note that the dynamics $\dot{x}_L$ and $\dot{\zeta}$ are of comparable size. Thus, we may use a single constant β and exponent γ for the exponential decay of both $x_L(t)$ and $\zeta(t)$ in order to simplify notation.

Consider the following cost function associated with system (5.1):

$$J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}\right) = \int_0^\infty (x_L^T Q x_L + x_L^T \mu_N^T R \mu_N x_L) dt \tag{5.11}$$

evaluated at a value $[x_{N0}^T, \ \ x_{L0}^T]^T$. The integral is taken along the trajectory which results from initial condition $[x_{N0}^T, \ \ x_{L0}^T]^T$. The weight factors Q and R are positive definite. They decompose into

$$Q = Q_1^T Q_1, \qquad R = R_1^T R_1, \tag{5.12}$$

where Q_1 and R_1 are square matrices of the appropriate dimensions. From exponential stability, it follows that $J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}\right)$ is finite for all $[x_{N0}^T, \ \ x_{L0}^T]^T$ belonging to X_0. From differentiability of the dynamics, and in conjuction with Proposition 2.2.1, it follows that the directional derivatives of J_μ along any direction exist.

Remark: In the cost function (5.11) there is no term involving the scheduling variables x_N. Therefore, the resulting controllers do not aim at forcing x_N to converge to 0. This is the case with other approaches [50] for gain scheduled control and is motivated by the fact that in many practical applications we are only interested in controlling the variables x_L and the scheduling variables x_N are simply viewed as exogenous factors which make the task of controlling x_L complicated. In contrast to this paradigm, the problem formulation of [47, 46] aims at controlling both x_N and x_L. The latter approach is more general and is motivated by practical problems. Such is the missile control problem, which is studied in Chapter 7. Omitting a term that weighs x_N in the cost function here makes easier the derivation of the continuity result in Section 5.2. However, the main hypothesis of this chapter, namely, that an architecture which is quadratic in x_L is a potentially good approximation strategy for the class of systems considered, is arguably valid even if such a term is included, as discussed in Section 5.3.

In the sequel, we consider a certain fixed initial condition $[x_{N0}^T, \quad x_{L0}^T]^T$ in X_0 of (5.1) at time $t = 0$, and let $[x_N^T(t), \quad x_L^T(t)]^T$ be the resulting trajectory. Consider the linear time invariant system

$$\dot{\zeta} = A_L(x_{N0})\zeta + g_L(x_{N0})\mu_N(x_{N0})\zeta, \quad \text{where} \quad \|x_{N0}\| \leq r_N \tag{5.13}$$

as in (5.9). Let J_{LTI} be the cost function associated with (5.13),

$$J_{LTI}(\zeta_0) = \int_0^\infty (\zeta^T Q \zeta + \zeta^T \mu_N^T(x_{N0}) R \mu_N(x_{N0})\zeta) dt. \tag{5.14}$$

evaluated at a value ζ_0. Notice that J_{LTI} also depends on x_{N0}, since the dynamics (5.13) depend on x_{N0}; however, this dependence is omitted for simplicity. The integral is taken along the trajectory of (5.13) which results from initial condition ζ_0. From exponential stability, $J_{LTI}(\zeta_0)$ is finite for every ζ_0 in $\mathbf{R}^{n-k}$. It is also differentiable, by virtue of Proposition 2.2.1.

5.2 A continuity result

Before stating the main result of the chapter, we state a result from the theory of ordinary differential equations which is useful in the proof of the main result:

Theorem 5.2.1 ([41], Theorem 2.5, p.79)

Let $f(t, x)$ be a piecewise continuous function in t that satisfies a Lipschitz condition *in x on $[0,T] \times W$ with a Lipschitz constant l, where $W \subset \mathbf{R}^n$ is an open connected set. Let $y(t)$ and $z(t)$ be solutions of*

$$\dot{y} = f(t,y), \quad y(0) = y_0$$

and

$$\dot{z} = f(t,z) + g(t,z), \quad z(0) = z_0$$

such that $y(t)$, $z(t) \in W$ for all $t \in [0,T]$. Suppose that

$$\|g(t,x)\| \leq \mu, \quad \forall t \in [0,T] \times W$$

for some $\mu > 0$, and

$$\|y_0 - z_0\| \leq \gamma.$$

Then,

$$\|y(t) - z(t)\| \leq \gamma e^{lt} + \frac{\mu}{l}(e^{lt} - 1)$$

for all $t \in [0,T]$.

We now state the following continuity result:

Proposition 5.2.1 *The difference*

$$L_{\begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix}} J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}\right) - L_{g_L(x_{N0})} J_{LTI}(x_{L0}) \tag{5.15}$$

between the directional derivatives of J_μ and J_{LTI} varies continuously with the size of the nonlinearity (expressed by Δ_i, $i = 1, \cdots, 12$).

As the size of the nonlinearity goes to 0, the difference is small and asymptotically *collapses to 0.*

5.3 Discussion of the Proposition

Proposition 5.2.1 indicates that a function which is quadratic in x_{L0} (J_{LTI}) is a good approximation to the cost of the nonlinear system, if the nonlinearity is small. The argument is clearly true if the dynamics are linear. If a small amount of nonlinearity is introduced, continuity implies that deviation from a function which is quadratic in x_{L0} is small. Therefore, an architecture which is quadratic in x_{L0} is a good cost function approximation.

As we take the limits $\Delta_i \to 0$, it follows that $x_N(t) \simeq x_{N0}$ for practically all t. It then looks like $f_L(x_N(t)) \simeq f_L(x_{N0})$ may not be close to 0, and a nonzero value of the control is needed in order to ensure that the lower equation of (5.4) is equal to 0 (since it is assumed that x_L asymptotically converges to 0). In consequence, the cost function would not be finite. It turns out that this is not a legitimate concern. Indeed, notice that, as the limit $\Delta_5 \to 0$ is taken, and by virtue of the nonlinear dynamics (5.4) Lipschitz property (Lemma 5.1.1, part 1), the initial value x_{N0} must be near points in the state space such that $||f_L||$ is 0. One such point is the origin (although there may be others too). It is then clear that the control is near 0 when x_L is close to the origin.

In the case where the only point where f_L is 0 is the origin, we could have included an additional term that weighs the deviation of the scheduling variables x_N from 0 in the cost function without essentially changing the result. We omitted doing so for simplicity.

Example: We present an example of a simple, academic nonlinear system of the form (5.4) and demonstrate that the directional derivative of J_μ is approximated very well by the directional derivative of J_{LTI}, in the sense of Proposition 5.2.1. The latter derivative is a linear function of x_{L0}. We consider the two-state system:

$$\begin{aligned}\dot{x}_N &= -x_N(\cosh(x_N)-1)+(cosh(x_N)-1)x_L\\ \dot{x}_L &= -x_N(\cosh(x_N)-1)-\frac{1}{2}(x_N^2+1)x_L-\frac{1}{2}(x_N^2+1)u\\ u &= x_L,\end{aligned}$$

where x_N, x_L and the input u are scalars. It can be easily verified that the above system is asymptotically stable with a Lyapunov function $V = x_N^2 + x_L^2$. The cost function $J_\mu([x_{N0}, x_{L0}]^T)$ is the integral $\int_0 \infty(x_L^2+u^2)dt = \int_0 \infty(2x_L^2)dt$ computed along the trajectory that starts off at $[x_{N0}, x_{L0}]^T$. We compute the directional derivatives of J_μ along the input vector $[0, -\frac{1}{2}(x_{N0}^2+1)]^T$ for varying values of the state $[x_{N0}, x_{L0}]^T$, where x_{N0} is kept fixed and x_{L0} varies as -5,-4,-3,. . .,5. We repeat the process for x_{N0}=0.1, 1, 5.

Notice that, as x_{N0} becomes small, the Δ_i bounds of the nonlinear dynamics become small.

The numerical computation of the directional derivatives is performed via integration of an appropriate system of differential equations, as described in detail later in the thesis, in Section 6.1.2. For each of the three values of x_{N0} we formulate the appropriate Linear Time Invariant system of the form (5.13) with the corresponding cost function J_{LTI}. For each fixed value of x_{N0} of 0.1, 1 and 5, we compute the directional derivatives of each system for values of x_{L0} varying as -5,-4,-3,. . .,5, and plot them against the corresponding directional derivatives of J_μ as they vary for varying x_{L0}. The plots are included in Figures 5.1, 5.2 and 5.3 for x_{N0} =0.1, 1, 5 respectively.

From these plots we see that the directional derivative of the nonlinear system as a function of x_{L0} is approximated very well for small values of x_{N0} (corresponding to small values of the size of the dynamics nonlinearities, as expressed by Δ_i's). As the size of x_{N0}, and therefore the size of the nonlinearity, increases, then the linear ap-

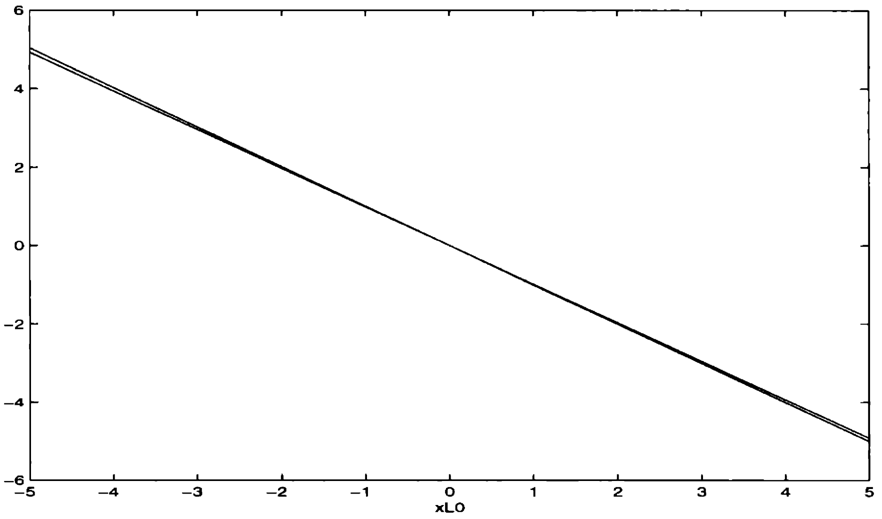

Figure 5.1: $x_{N0} = 0.1$ and x_{L0} varying between -5 and 5: the directional derivative of J_μ as a function of x_{L0} versus the directional derivative of J_{LTI} as a function of x_{L0}

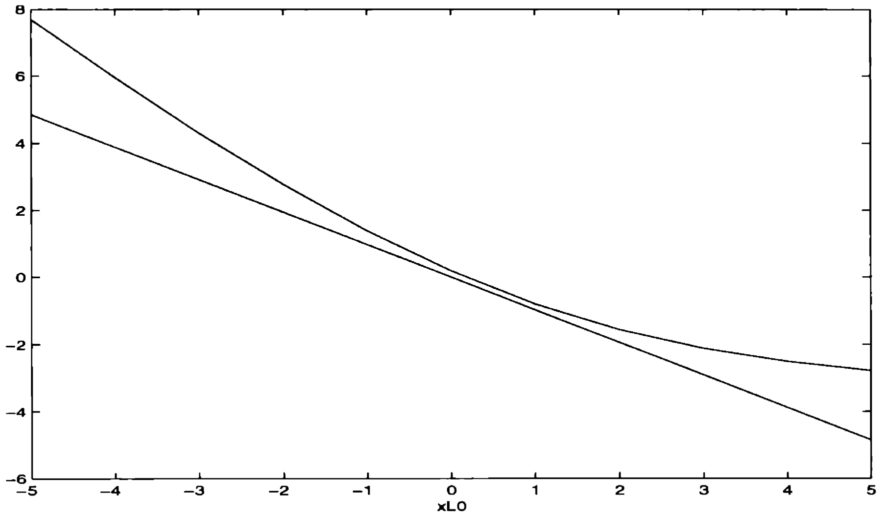

Figure 5.2: $x_{N0} = 1$ and x_{L0} varying between -5 and 5: the directional derivative of J_μ as a function of x_{L0} versus the directional derivative of J_{LTI} as a function of x_{L0}

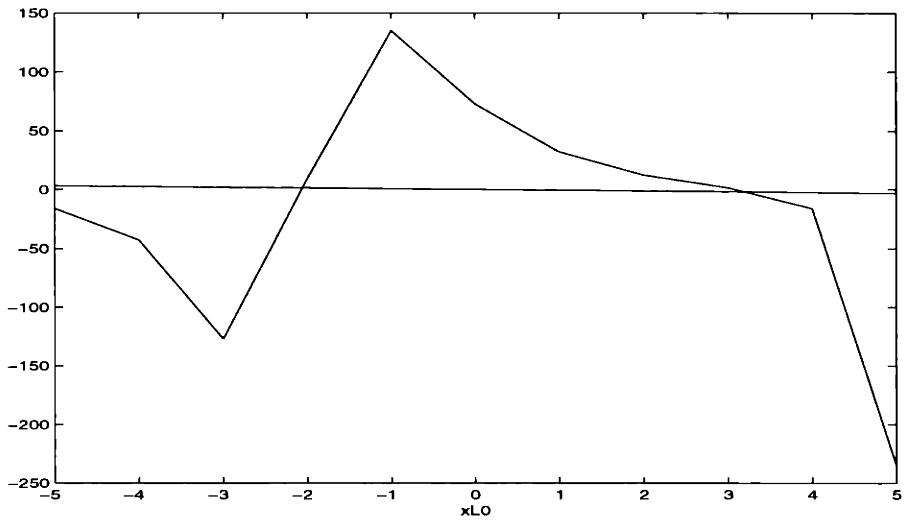

Figure 5.3: $x_{N0} = 5$ and x_{L0} varying between -5 and 5: the directional derivative of J_μ as a function of x_{L0} versus the directional derivative of J_{LTI} as a function of x_{L0}

proximation is not very good. At large values of the size of the nonlinearity (x_{N0}=5), the approximation is totally inaccurate. However, the linear approximation is valid for a large range of x_{N0}, as demonstrated in Figure 5.2 for x_{N0}=1. Therefore, for systems of the form (5.4) it is always recommendable to try a cost function approximation architecture which is quadratic in the variables x_L (which translates into a linear dependence of the directional derivative of J_μ on x_L.

5.4 Proof of the Proposition

Throughout the proof, we define several terms denoted D_i, $i = 1, \ldots, 11$, in order to simplify the formulas. In each definition we use the symbol $\triangleq$, so that they are recognized by the reader.

Consider a certain point $\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}$ in X_0. We want to find an upper bound on the absolute value of the difference between the two directional derivatives:

$$\left| \lim_{\delta \to 0} \frac{J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix}\right) - J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}\right) - J_{LTI}(x_{L0} + \delta g_L(x_{N0})) + J_{LTI}(x_{L0})}{\delta} \right| . \tag{5.16}$$

Since $\delta \to 0$, we only need to consider δ small enough such that $\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix}$ belongs to X_0.

Given a certain point $\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}$ in X_0, we denote by $\begin{bmatrix} x_N(t) \\ x_L(t) \end{bmatrix}$ and $\begin{bmatrix} \bar{x}_N(t) \\ \bar{x}_L(t) \end{bmatrix}$ the trajectories that satisfy (5.4) and the initial conditions

$$\begin{bmatrix} x_N(0) \\ x_L(0) \end{bmatrix} = \begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} \text{ and } \begin{bmatrix} \bar{x}_N(0) \\ \bar{x}_L(0) \end{bmatrix} = \begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix}$$

respectively. Similarly, we denote by $z(t)$ and $\bar{z}(t)$ the trajectories that satisfy (5.13) and the initial conditions

$$x_{L0} \quad \text{and } x_{L0} + \delta g_L(x_{N0})$$

respectively.

Step 1.

To start with the proof, we are showing an important property of the cost integrals, which stems from the exponential stability Assumption 5.1.5, and which is very useful for the rest of the proof.

Lemma 5.4.1 *If Assumption 5.1.5 holds, then:*

1. *The accumulated cost from some time $T > 0$ up to ∞ along any trajectory of*

system (5.4) is upper bounded by

$$\frac{C\beta^2}{\gamma}e^{-2\gamma T}, \tag{5.17}$$

where

$$C \triangleq \frac{1}{2}(\|Q\|r_L^2 + \|R\|r_N^2 r_L^2 l_\mu) \tag{5.18}$$

2. *The accumulated cost from some time $T > 0$ up to ∞ along any trajectory of system (5.13), is upper bounded by*

$$\frac{C\beta^2}{\gamma}e^{-2\gamma T}, \tag{5.19}$$

Proof of Lemma 5.4.1:

1. Let us consider any trajectory $[x_N^T(t),\ \ x_L^T(t)]$, starting from $[x_{N0}^T,\ \ x_{N0}^T]$ at time $t = 0$. The accumulated cost

$$\int_T^\infty \left(x_L^T Q x_L + x_L^T \mu_N(x_N)^T R \mu_N(x_N) x_L\right) dt$$

is upper bounded by

$$\int_T^\infty x_L^T Q x_L dt + \int_T^\infty x_L^T \mu_N(x_N)^T R \mu_N(x_N) x_L dt.$$

By virtue of Assumptions 5.1.1 and 5.1.5, Lemma 5.1.1, and norm properties, the above sum of integrals is upper bounded by

$$\begin{array}{cc}
\leq & \int_T^\infty \|Q\|\|x_L\|^2 dt + \int_T^\infty \|R\|\|\mu_N(x_N)\|^2\|x_L\|^2 dt \\
\leq & \int_T^\infty \|Q\|\beta^2\|x_{L0}\|^2 e^{-2\gamma t}dt + \int_T^\infty \|R\|l_\mu^2\|x_N\|^2\beta^2\|x_{L0}\|^2 e^{-2\gamma t}dt \\
\leq & \beta^2\left(\|Q\|r_L^2 + \|R\|r_L^2 r_N^2 l_\mu^2\right)\frac{1}{2\gamma}e^{-2\gamma T}
\end{array}$$

2. The proof follows in the same way as the proof of Part 1. ∎

Step 2.

We define the following signals:

$$\begin{bmatrix} y_N(t) \\ y_L(t) \end{bmatrix} \triangleq \frac{1}{\delta}\left(\begin{bmatrix} \bar{x}_N(t) \\ \bar{x}_L(t) \end{bmatrix} - \begin{bmatrix} x_N(t) \\ x_L(t) \end{bmatrix}\right) \tag{5.20}$$

$$w(t) \triangleq \frac{1}{\delta}(\bar{z}(t) - z(t)) \tag{5.21}$$

By differentiating (5.20) and (5.21) we obtain

$$\begin{bmatrix} \dot{\bar{x}}_N(t) \\ \dot{\bar{x}}_L(t) \end{bmatrix} = \begin{bmatrix} \dot{x}_N(t) \\ \dot{x}_L(t) \end{bmatrix} + \delta \begin{bmatrix} \dot{y}_N(t) \\ \dot{y}_L(t) \end{bmatrix} \tag{5.22}$$

$$\dot{\bar{z}}(t) = \dot{z}(t) + \delta \dot{w}(t). \tag{5.23}$$

We write the dynamics of $[\bar{x}_N^T,\ \bar{x}_L^T]^T$:

$$\begin{bmatrix} \dot{\bar{x}}_N \\ \dot{\bar{x}}_L \end{bmatrix} = \begin{bmatrix} f_N(\bar{x}_N) \\ f_L(\bar{x}_N) \end{bmatrix} + \begin{bmatrix} A_N(\bar{x}_N) \\ A_L(\bar{x}_N) \end{bmatrix} \bar{x}_L + \begin{bmatrix} g_N(\bar{x}_N) \\ g_L(\bar{x}_N) \end{bmatrix} \mu_N(\bar{x}_N)\bar{x}_L \tag{5.24}$$

By using (5.20), and by neglecting the δ^2 terms, which converge to 0 faster than δ as $\delta \to 0$, we obtain from (5.24):

$$\begin{aligned} \begin{bmatrix} \dot{\bar{x}}_N \\ \dot{\bar{x}}_L \end{bmatrix} &= \begin{bmatrix} f_N(x_N) + \delta \frac{\partial f_N(x_N)}{\partial x_N} y_N \\ f_L(x_N) + \delta \frac{\partial f_L(x_N)}{\partial x_N} y_N \end{bmatrix} + \begin{bmatrix} A_N(x_N) \\ A_L(x_N) \end{bmatrix} \bar{x}_L + \delta \begin{bmatrix} \sum_{i=1}^k (\frac{\partial A_N(x_N)}{\partial x_{N_i}} y_{N_i}) \\ \sum_{i=1}^k (\frac{\partial A_L(x_N)}{\partial x_{N_i}} y_{N_i}) \end{bmatrix} \bar{x}_L + \\ &+ \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} \mu_N(x_N)\bar{x}_L + \delta \begin{bmatrix} \sum_{i=1}^k \frac{\partial g_N}{\partial x_{N_i}} y_{N_i} \\ \sum_{i=1}^k \frac{\partial g_L}{\partial x_{N_i}} y_{N_i} \end{bmatrix} \mu_N(x_N)\bar{x}_L + \\ &+\delta \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} \left(\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i}\right) \bar{x}_L \end{aligned}$$

$$= \begin{bmatrix} \dot{x}_N \\ \dot{x}_L \end{bmatrix} + \delta \left\{ \begin{bmatrix} \frac{\partial f_N(x_N)}{\partial x_N} y_N \\ \frac{\partial f_L(x_N)}{\partial x_N} y_N \end{bmatrix} + \begin{bmatrix} A_N(x_N) \\ A_L(x_N) \end{bmatrix} y_L + \begin{bmatrix} \sum_{i=1}^k \left(\frac{\partial A_N(x_N)}{\partial x_{N_i}} y_{N_i} \right) \\ \sum_{i=1}^k \left(\frac{\partial A_L(x_N)}{\partial x_{N_i}} y_{N_i} \right) \end{bmatrix} x_L + \right.$$

$$+ \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} \mu_N(x_N) y_L + \begin{bmatrix} \sum_{i=1}^k \frac{\partial g_N}{\partial x_{N_i}} y_{N_i} \\ \sum_{i=1}^k \frac{\partial g_L}{\partial x_{N_i}} y_{N_i} \end{bmatrix} \mu_N(x_N) x_L +$$

$$\left. + \begin{bmatrix} g_N(x_N) \\ g_L(x_N) \end{bmatrix} \left(\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} \right) x_L \right\}. \tag{5.25}$$

Let us denote the expression inside the curly brackets by a vector function $\begin{bmatrix} Y_N(x_N, x_L, y_N, y_L) \\ Y_L(x_N, x_L, y_N, y_L) \end{bmatrix}$. From (5.22) and (5.25), it turns out that the dynamics of y_N and y_L are given by

$$\begin{bmatrix} \dot{y}_N \\ \dot{y}_L \end{bmatrix} = \begin{bmatrix} Y_N(x_N, x_L, y_N, y_L) \\ Y_L(x_N, x_L, y_N, y_L) \end{bmatrix}. \tag{5.26}$$

Similarly, we get

$$\dot{\bar{z}} = A_L(x_{N0})z + g_L(x_{N0})\mu_N(x_{N0})z + \delta(A_L(x_{N0})w + g_L(x_{N0})\mu_N(x_{N0})w) \tag{5.27}$$

from which, in conjuction with (5.23), we obtain the dynamics of w as

$$\dot{w} = (A_L(x_{N0}) + g_L(x_{N0})\mu_N(x_{N0}))\, w \tag{5.28}$$

Step 3. We apply Theorem 5.2.1 on the signals $x(t)$ and $\bar{x}(t)$, both of which obey (5.3), but start at initial conditions that differ by $\delta[g_N^T(x_{N0}),\ \ g_L^T(x_{N0})]^T$. We define $g_c \triangleq \|[g_N^T(x_{N0}),\ \ g_L^T(x_{N0})]^T\|$ and by virtue of Theorem 5.2.1 we conclude that, for every $T > 0$

$$\left\| \begin{bmatrix} \bar{x}_N(t) \\ \bar{x}_L(t) \end{bmatrix} - \begin{bmatrix} x_N(t) \\ x_L(t) \end{bmatrix} \right\| = \delta \cdot \left\| \begin{bmatrix} y_N \\ y_L \end{bmatrix} \right\| \leq \delta \left\| \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix} \right\| e^{l_{NL}T} = \delta g_c e^{l_{NL}T}$$

$$\Rightarrow \qquad \|y_N(t)\| \leq g_c e^{l_{NL}T} \quad \text{and} \quad \|y_L(t)\| \leq g_c e^{l_{NL}T} \tag{5.29}$$

for every $t \in [0, T]$. In a similar way, we argue by virtue of Theorem 5.2.1 that

$$\|w(t)\| \leq \|g_L(x_{N0})\| e^{l_{LTI}T} \leq g_c e^{l_{LTI}T} \tag{5.30}$$

for every $T > 0$, for all $t \in [0, T]$, where l_{LTI} is a Lipschitz constant of system (5.13).

Step 4. We consider the cost integral J_μ. We use definition (5.20) to write $\bar{x}_N$ and $\bar{x}_L$, and we drop the terms including δ^2 factors, for $\delta \to 0$.

$$J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix} \right) = \int_0^\infty (\bar{x}_L^T Q \bar{x}_L + \bar{x}_L^T \mu_N(\bar{x}_N)^T R \mu_N(\bar{x}_N) \bar{x}_L) dt =$$

$$= \int_0^\infty \{x_L^T Q x_L + 2\delta y_L^T Q x_L +$$

$$+(x_L + \delta y_L)^T (\mu_N(x_N) + \delta \sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i})^T R (\mu_N(x_N) + \delta \sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i})(x_L + \delta y_L)\} dt =$$

$$= \int_0^\infty (x_L^T Q x_L + x_L^T \mu_N(x_N)^T R \mu_N(x_N) x_L) dt +$$

$$+2\delta \int_0^\infty \{y_L^T Q x_L + x_L^T \mu_N(x_N)^T R \sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L + y_L^T \mu_N^T(x_N) R \mu_N(x_N) x_L\} dt. \tag{5.31}$$

It follows from (5.31) that

$$\frac{1}{\delta}\left\{ J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_L(x_{N0}) \end{bmatrix} \right) - J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} \right) \right\} =$$

$$= 2\int_0^\infty \{y_L^T(Q + \mu_N(x_N)^T R\mu_N(x_N))x_L + x_L^T\mu_N(x_N)^T R\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L\}dt \quad (5.32)$$

Similarly it can be shown that

$$\frac{1}{\delta}\{J_{LTI}(x_{L0} + \delta g_L(x_{N0})) - J_{LTI}(x_{L0})\} = 2\int_0^\infty (w^T(Q + \mu_N(x_{N0})^T R\mu_N(x_{N0}))z)dt \quad (5.33)$$

Our objective for the rest of the proof is to derive an upper bound on the difference

$$2\int_0^\infty (y_L^T(Q + \mu_N(x_N)^T R\mu_N(x_N))x_L + x_L^T\mu_N(x_N)^T R\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L)dt - \\ -2\int_0^\infty (w^T(Q + \mu_N(x_{N0})^T R\mu_N(x_{N0}))z)dt \quad (5.34)$$

and show that the bound collapses to 0 as $\Delta \to 0$.

Step 5.

It turns out from Lemma 5.4.1 that

$$\left|\frac{1}{\delta}\left(\int_T^\infty (\bar{x}_L^T Q\bar{x}_L + \bar{x}_L^T\mu_N(\bar{x}_N)^T R\mu_N(\bar{x}_N)\bar{x}_L)dt \right.\right. \\ \left.\left. - \int_T^\infty (x_L^T Q x_L + x_L^T\mu_N(x_N)^T R\mu_N(x_N)x_L)dt\right)\right| \\ \leq \frac{2C\beta^2}{\delta\gamma}e^{-2\gamma T}. \quad (5.35)$$

Thus, the rate at which the difference between the accumulated costs from $t = T$ up to ∞ along $\bar{x}$ and x is decaying exponentially with T. Since such bounds exist for every δ, it follows from differentiability of J_μ that if we take the limit of the left side of (5.35) as $\delta \to 0$, the limit exists and it decays exponentially in T. In other words, there exists a function $C_{NL}(\delta)$ such that the left hand side of inequality (5.35) is upper bounded by

$$\frac{C_{NL}(\delta)\beta^2}{\delta\gamma}e^{-2\gamma T}$$

and

$$\lim_{\delta \to 0} \frac{C_{NL}(\delta)}{\delta} = \bar{C}_{NL},$$

where $\bar{C}_{NL}$ is a well defined real number. In conjuction with (5.31), it follows that, for every $T > 0$,

$$\left| 2 \int_T^\infty (y_L^T (Q + \mu_N(x_N)^T R \mu_N(x_N)) x_L + x_L^T \mu_N(x_N) R \sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L) dt \right| \leq \frac{C_{NL}(\delta)\beta^2}{\delta\gamma} e^{-2\gamma T} \tag{5.36}$$

$$\text{where} \quad \lim_{\delta \to 0} \frac{C_{NL}(\delta)}{\delta} = \bar{C}_{NL}. \tag{5.37}$$

Similarly, it can be argued that there exists a function $C_{LTI}(\delta)$ such that

$$\left| 2 \int_T^\infty (w^T (Q + \mu_N(x_{N0})^T R \mu_N(x_{N0})) z) dt \right| \leq \frac{C_{LTI}(\delta)\beta^2}{\delta\gamma} e^{-2\gamma T} \tag{5.38}$$

$$\text{where } \lim_{\delta \to 0} \frac{C_{LTI}(\delta)}{\delta} = \bar{C}_{LTI} \tag{5.39}$$

We now define the function $\epsilon(T)$ as follows:

Definition: Given a positive T, $\epsilon(T)$ is defined as

$$\epsilon(T) \triangleq \max \left\{ \frac{\bar{C}_{NL}\beta^2}{\gamma} e^{-2\gamma T}, \quad \frac{\bar{C}_{LTI}\beta^2}{\gamma} e^{-2\gamma T} \right\} \tag{5.40}$$

This definition will not be used again until Step 9.

Step 6.

We define $h(z) \triangleq A_L(x_{N0})z + g_L(x_{N0})\mu_N(x_{N0})z$ and consider the system:

$$\begin{aligned} \dot{z} &= A_L(x_{N0})z + g_L(x_{N0})\mu_N(x_{N0})z \\ &= h(z), \qquad z(0) = x_{L0}. \end{aligned} \tag{5.41}$$

We also define

$$h_p(x_L, t) \triangleq f_L(x_N(t)) + (A_L(x_N(t)) - A_L(x_{N0}))x_L + ((g_L\mu_N)(x_N(t)) - (g_L\mu_N)(x_{N0}))x_L \tag{5.42}$$

and consider the system:

$$\begin{aligned} \dot{x}_L &= f_L(x_N(t)) + A_L(x_N(t))x_L + g_L(x_N(t))\mu_N(x_N(t))x_L \\ &= h(x_L) + h_p(x_L, t), \qquad x(0) = x_{L0} \end{aligned} \tag{5.43}$$

In order to apply Theorem 5.2.1 on systems (5.41) and (5.43), we need to determine an upper bound on $\|h_p(x_L, t)\|$. From the dynamic equation of $\|x_N\|$ we obtain

$$x_N(T) - x_{N0} = \int_0^T (f_N(x_N) + A_N(x_N)x_L + (g_N\mu_N)(x_N)x_L)dt \tag{5.44}$$

It follows from (5.44) in conjuction with Lemma 5.1.1 that

$$\|x_N(T) - x_{N0}\| \le T(\Delta_1 + \Delta_2 r_L + \Delta_3 l_\mu r_N r_L) \triangleq D_1(T, \Delta) \tag{5.45}$$

From (5.45) and Lemma 5.1.1, we obtain:

$$\|h_p(x_L, t)\| \le \Delta_5 + l_{A_L} D_1(T, \Delta) + l_{g\mu} D_1(T, \Delta) \triangleq D_2(T, \Delta), \quad \text{for all } t \in [0, T] \tag{5.46}$$

Also,

$$\|h(z_1) - h(z_2)\| \leq \|A_L(x_{N0}) + g_L(x_{N0})\mu_N(x_{N0})\| \cdot \|z_1 - z_2\|. \tag{5.47}$$

Thus, the Lipschitz constant of h is given by $l_{LTI} = \|A_L(x_{N0}) + g_L(x_{N0})\mu_N(x_{N0})\|$. Then, it follows from Theorem 5.2.1 that:

$$\|z(t) - x_L(t)\| \leq \frac{D_2(T,\Delta)}{l_{LTI}}(e^{l_{LTI}T} - 1) \triangleq D_3(T,\Delta), \qquad \text{for all } t \in [0,T]. \tag{5.48}$$

From (5.48) we obtain

$$\begin{aligned} & \|z(t)\| - \|x_L\| \leq \|z(t) - x_L(t)\| \leq D_3(T,\Delta) \\ \Rightarrow \quad & \|z(t)\| \leq \|x_L\| + D_3(T,\Delta) \\ \Rightarrow \quad & \|z(t)\| \leq r_L + D_3(T,\Delta) \end{aligned} \tag{5.49}$$

Step 7.

We consider the dynamic system (5.28). We define $H(w) \triangleq (A_L(x_{N0}) + g_L(x_{N0})\mu_N(x_{N0}))\, w$, and we consider the trajectory

$$\dot{w} = H(w), \qquad w(0) = g_L(x_{N0}). \tag{5.50}$$

We define $G(y_L,t) \triangleq Y_L(x_N(t), x_L, y_N(t), y_L(t)) - H(y_L)$, and consider the trajectory:

$$\dot{y}_L = Y_L(x_N, x_L, y_N, y_L) = H(y_L) + G(y_L,t), \qquad y_L(0) = g_L(x_{N0}). \tag{5.51}$$

From (5.26) and (5.25), it turns out that

$$G(y_L,t) = \frac{\partial f_L(x_N)}{\partial x_N}(t)y_N(t) + (A_L(x_N(t)) - A_L(x_{N0}))y_L + \sum_{i=1}^{k}\left(\frac{\partial A_L(x_N)}{\partial x_{N_i}}y_{N_i}\right)x_L +$$

$$
\begin{aligned}
&+ \left(g_L(x_N(t))\mu_N(x_N) - g_L(x_{N0})\mu_N(x_{N0})\right) y_L + \\
&+ \left(\sum_{i=1}^{k} \frac{\partial g_L}{\partial x_{N_i}} y_{N_i}\right) \mu_N(x_N) x_L + g_L \left(\sum_{i=1}^{k} \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i}\right) x_L
\end{aligned}
\tag{5.52}
$$

Based on Lemma 5.1.1 and inequalities (5.45) and (5.29), it turns out that $||G(y_L, t)||$ is upper bounded, as

$$
\begin{aligned}
||G(y_L, t)|| &\leq \Delta_7 ||y_N(t)|| + l_{A_L} ||x_N(t) - x_{N0}|| ||y_N(t)|| + \Delta_9 ||y_N(t)|| ||x_L|| + \\
&\quad + l_{g\mu} ||x_N(t) - x_{N0}|| ||y_L(t)|| \\
&\quad + \Delta_{11} ||y_N(t)|| l_\mu r_N ||x_L|| + \Delta_4 \Delta_{12} ||y_N|| ||x_L|| ||g_L(0)|| \Delta_{12} ||y_N|| ||x_L|| \\
&\leq (\Delta_7 + l_{A_L} D_1(T, \Delta) + \Delta_9 r_L + l_{g\mu} D_1(T, \Delta) \\
&\quad + \Delta_{11} l_\mu r_N r_L + \Delta_4 \Delta_{12} r_L + ||g_L(0)|| \Delta_{12} r_L)\, g_c e^{l_{NL} T} \\
&\triangleq D_4(T, \Delta), \qquad \text{for all } \; t \in [0, T].
\end{aligned}
\tag{5.53}
$$

By applying Theorem 5.2.1 on (5.50) and (5.51), we obtain:

$$
||y_L(t) - w(t)|| \leq \frac{D_4(T, \Delta)}{l_{LTI}} (e^{l_{LTI} T} - 1) \triangleq D_5(T, \Delta) \qquad \text{for all } \; t \in [0, T]. \tag{5.54}
$$

As a direct consequence of (5.54) and (5.29), we have for every $t \in [0, T]$:

$$
\begin{aligned}
& ||w(t)|| - ||y_L(t)|| \leq ||y_L(t) - w(t)|| \leq D_5(T, \Delta) \\
\Rightarrow \quad & ||w(t)|| \leq ||y_L(t)|| + D_5(T, \Delta) \\
\Rightarrow \quad & ||w(t)|| \leq g_c e^{l_{NL} T} + D_5(T, \Delta)
\end{aligned}
\tag{5.55}
$$

Step 8.

The next step is to show that the difference

$$2\int_0^T \left[y_L^T \left(Q + \mu_N(x_N)^T R \mu_N(x_N) \right) x_L + x_L^T \mu_N(x_N)^T R \left(\sum_{i=1}^{k} \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} \right) x_L \right] dt - $$
$$-2\int_0^T \left(w^T (Q + \mu_N^T(x_{N0}) R \mu_N(x_{N0}) \right) dt \qquad (5.56)$$

is small. We have:

$$\begin{aligned}
2\left|\int_0^T \left(y_L^T Q x_L - w^T Q z \right) dt \right| &= \left| \int_0^T \left(2 y_L^T Q_1^T Q_1 x_L - 2 w^T Q_1^T Q_1 z \right) dt \right| \\
&= \Big| \int_0^T \Big((y_L^T Q_1^T - w^T Q_1^T)(Q_1 x_L + Q_1 z) + \\
&\quad + (y_L^T Q_1^T + w^T Q_1^T)(Q_1 x_L - Q_1 z) \Big) dt \Big| \\
&= \left| \int_0^T \left((y_L^T - w^T) Q (x_L + z) + (y_L^T + w^T) Q (x_L - z) \right) dt \right| \\
&\leq \left| (y_L^T - w^T) Q (x_L + z) + (y_L^T + w^T) Q (x_L - z) \right| T \\
&\leq [\|(y_L - w)\| \|Q\| (\|x_L\| + \|z\|) + (\|y_L\| + \|w\|) \|Q\| \|x_L - z\|] \, T \\
&\leq [D_5(T,\Delta) \|Q\| (2 r_L + D_3(T,\Delta)) + \\
&\quad + (2 g_c e^{l_{NL} T} + D_5(T,\Delta)) \|Q\| D_3(T,\Delta) \big] \, T \\
&\triangleq D_6(T,\Delta) \qquad (5.57)
\end{aligned}$$

In a similar manner to (5.57), we obtain

$$\begin{aligned}
& 2\left| \int_0^T \left(y_L^T \mu_N(x_N)^T R \mu_N(x_N) x_L - w^T \mu_N(x_N)^T R \mu_N(x_N) z \right) dt \right| \leq \\
& \leq \left| (y_L^T - w^T) \mu_N(x_N)^T R \mu_N(x_N)(x_L + z) + (y_L^T + w^T) \mu_N(x_N)^T R \mu_N(x_N)(x_L - z) \right| T \\
& \leq \left(\|y_L - w\| \|\mu x_N(x_N)\|^2 \|R\| (\|x_L\| + \|z\|) + (\|y_L\| + \|w\|) \|\mu_N(x_N)\|^2 \|R\| \|x_L - z\| \right) \\
& \leq \left(D_5(T,\Delta) l_\mu^2 r_N^2 \|R\| (2 r_L + D_3(T,\Delta)) + (2 g_c e^{l_{NL} T} + D_5(T,\Delta)) l_\mu^2 r_N^2 \|R\| D_3(T,\Delta) \right) T \\
& \triangleq D_7(T,\Delta) \qquad (5.58)
\end{aligned}$$

Finally,

$$\begin{aligned} 2\left|\int_0^T x_L^T \mu_N(x_N)^T R\left(\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L\right) dt\right| &\leq 2\|x_L\|^2 l_\mu r_N \|R\| \Delta_{12} \|y_N\| T \\ &\leq 2 r_L^2 l_\mu r_N \|R\| \Delta_{12} g_c e^{l_{NL}T} T \\ &\triangleq D_9(T,\Delta) \qquad (5.59) \end{aligned}$$

We define

$$D(T,\Delta) \triangleq D_6(T,\Delta) + D_7(T,\Delta) + D_8(T,\Delta) + D_9(T,\Delta) \qquad (5.60)$$

to conclude that

$$\begin{aligned} &\left|2\int_0^T \left[y_L^T\left(Q + \mu_N(x_N)^T R \mu_N(x_N)\right) x_L + x_L^T \mu_N(x_N)^T R\left(\sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i}\right) x_L\right] dt - \right. \\ &\left. -2\int_0^T \left(w^T(Q + \mu_N^T(x_{N0}) R \mu_N(x_{N0})\right) dt\right| \leq D(T,\Delta) \qquad (5.61) \end{aligned}$$

Step 9.

Shorthand notation:

To keep the rest of the proof short, we are introducing the notation $I(t)$ for the difference between the two integrands in (5.34), (5.56),

$$\begin{aligned} I(t) \triangleq\ & 2(y_L^T(Q + \mu_N(x_N)^T R \mu_N(x_N)) x_L + x_L^T \mu_N(x_N)^T R \sum_{i=1}^k \frac{\partial \mu_N}{\partial x_{N_i}} y_{N_i} x_L) - \\ & -2(w^T(Q + \mu_N(x_{N0})^T R \mu_N(x_{N0}))z) \qquad (5.62) \end{aligned}$$

We can write:

$$\int_0^\infty I(t)dt = \int_0^T I(t)dt + \int_T^\infty I(t)dt \qquad (5.63)$$

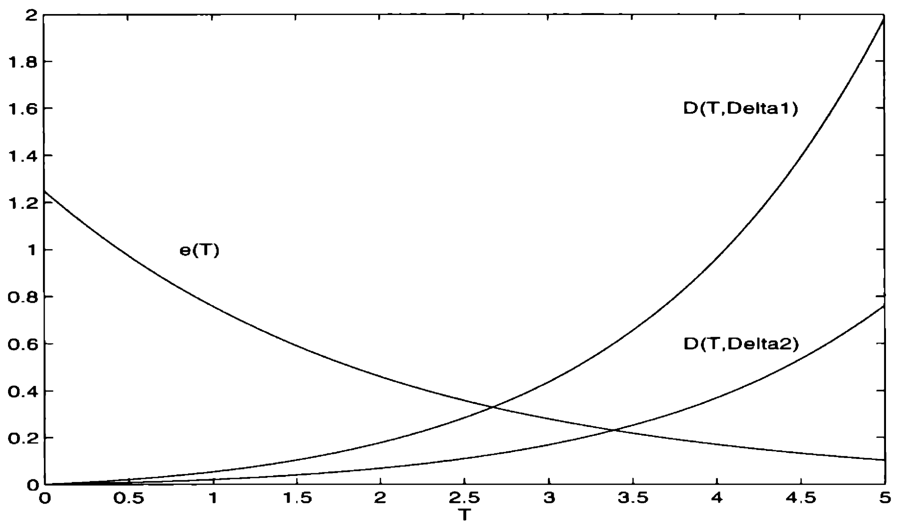

Figure 5.4: The functions $\epsilon(T)$ and $D(T,\Delta)$ for two values $\Delta_1 > \Delta_2$ of Δ

From the definition (5.40) of the function $\epsilon(T)$ and from (5.61), it follows that

$$|\int_0^\infty I(t)dt| \leq \epsilon(T) + D(T,\Delta) \tag{5.64}$$

Consider the two functions $\epsilon(T)$ and $D(T,\Delta)$, for a fixed value of Δ. For T ranging from 0 to ∞, the function $\epsilon(T)$ ranges from a finite value $\epsilon(0)$ given by

$$\epsilon(0) = \min\left\{\frac{1}{\delta}\left[J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix} + \delta \begin{bmatrix} g_N(x_{N0}) \\ g_N(x_{N0}) \end{bmatrix}\right) - J_\mu\left(\begin{bmatrix} x_{N0} \\ x_{L0} \end{bmatrix}\right)\right], \right.$$
$$\left. \frac{1}{\delta}\left\{J_{LTI}(x_{L0} + \delta g_L(X_{N0})) - J_{LTI}(x_{L0})\right\}\right\} \tag{5.65}$$

to $\lim_{T\to\infty}\epsilon(T) = 0$. At the same time, $D(T,\Delta)$ ranges from $D(0,\Delta) = 0$ to ∞, as T ranges from 0 to ∞. The two continuous curves intersect at some point $T_s(\Delta)$ such that $\epsilon(T_s(\Delta)) = D(T_s(\Delta),\Delta)$. Therefore,

$$|\int_0^\infty I(t)dt| \leq 2D(T_s(\Delta),\Delta) \tag{5.66}$$

Step 10.

Let us now examine what happens at the limit $||\Delta|| \rightarrow 0$. From what we have seen so far, the function $\epsilon(T)$ is independent of the size of Δ (more precisely, it is upper bounded by a fixed function of Δ, see (5.36) and (5.38)). However, for any fixed value of T, the function $D(T, \Delta)$ ranges from 0 as $||\Delta|| \rightarrow 0$ to ∞, as $||\Delta||$ grows unbounded. See Figure 5.4. As $||\Delta||$ decreases, the curve $D(T, \Delta)$ shifts downwards. Thus, it intersects with the $\epsilon(T)$ curve closer to 0, and actually converges to 0 as $||\Delta|| \rightarrow 0$

■

Chapter 6

A Grid-based Approximation Architecture

In Chapters 2, 3 and 4 we developed bounds on the approximation error under which approximate policy iteration is a valid tool for nonlinear control design. However, success is not guaranteed, unless these error bounds are respected. Let us recall the way suggested so far in the thesis for cost function approximation, so as to motivate the alternative approximation architecture proposed in this chapter. By an approximation architecture we refer to a set of functions mapping $\mathbf{R}^n$ to $\mathbf{R}$ which is parametrized by a collection of scalar variables, $r_1, \cdots, r_m$. The values of these variables are selected so that the resulting function (among all functions belonging to the architecture), $\tilde{J}_{\mu_k}$ best matches the actual cost function $J^c_{\mu_k}$. For example, so far in this thesis we have considered architectures of the form (1.12),

$$r_1 h_1(x) + \cdots + r_m h_m(x), \tag{6.1}$$

These are linear combinations of a collection of nonlinear functions $h_1, \cdots, h_m$. As shown in Chapter 1, for each policy μ_k, the values of r_i are selected so as to minimize the least square error between the architecture and the actual cost function, based on a sample of the latter (1.13). There are a couple of advantages associated with an architecture of this type. First, the designer can choose the nonlinear functions at will, and thus use his/her skills and insight in order to produce a simple approximation function. Second, the architecture is linear with respect to the free parameters r_i, and therefore some computationally easy ways can be used in order to "tune" the latter so that the actual cost function is best matched, e.g. (1.13). At the same time, there is a potentially serious disadvantage to this approach in that there are no guarantees, other than the designer's ingenuity, that a good cost function approximation can be constructed for a given system. This problem can potentially be addressed via use of **systematic** approximation approaches. One such approach is via use of *neural networks.* The neural network architectures are nonlinear with respect to the tunable parameters r_i. They consist of a weighted chain of nonlinear functions, where r_i are the weights. It can be shown that, as the size of the neural network grows, any function can be approximated arbitrarily well. Among the disadvantages of a neural network is the difficulty of determining the minimum size such that the resulting architecture is acceptable for a given cost function. Another disadvantage is associated with the difficulty in computing the values of the weights r_i such that the matching criteria between the network architecture and the actual cost function are best met. Another systematic approach is via the use of an architecture of the type (6.1), where each $h_i(x)$ is a polynomial term with respect to the scalar elements of x. According to the multivariable Stone-Weirstrass theorem, if all polynomial terms of size up to L are included, and in the limit as $L \to \infty$, any function can be approximated arbitrarily closely by the architecture. As in the case of neural networks, determining the minimum value of L that results in an acceptable approximation is not an easy task.

In this chapter we propose an alternative systematic approximation architecture.

It is based on a grid of the state space. Given a policy μ, approximate policy iteration results in an updated controller μ'. If the cost function of the closed loop system under μ is denoted by J_μ^c, then μ' is proportional to the approximate directional derivative of J_μ^c in the direction of the input matrix G. Thus, we compute the directional derivative of J_μ^c at the vertices of the grid (via integration of a set of differential equations, as shown in Section 6.1), and we use affine interpolation to approximate the directional derivative throughout the rest of the state space. Among the advantages of the approach is simplicity and ease of implementation. Another advantage is that the criteria developed in Chapter 2 can be systematically checked for such an architecture. The disadvantage of the approach is the universal disadvantage of any systematic computational approach for approximating a function in a Euclidean state space, that is, dimensionality. For a given problem, any grid refinement aiming at improving approximation results in exponential growth of the requirements for computing the approximate cost function. Furthermore, computational requirements grow exponentially with the dimension of the state space. However, the advantages of the architecture over other systematic approximation strategies make it a very attractive alternative. In the rest of the chapter, we describe the architecture, and develop a convex optimization test for checking whether the stability criterion (2.29) is satisfied.

6.1 Approximation Architecture

6.1.1 Grid Selection

Suppose that we start with a system controlled by a policy μ,

$$\dot{x} = f(x) + G(x)\mu(x), \quad x(t=0) = x_0, \tag{6.2}$$

where $x \in \mathbf{R}^n$ and $\mu(x) \in \mathbf{R}^m$, such that the origin 0 is the only equilibrium point of (6.2). Assume that (6.2) is asymptotically stable and that $J_\mu^c(x_0)$ is finite for all x_0 belonging to a bounded region X_0 which includes the origin. Let the dynamics satisfy the continuous differentiability assumptions of Chapter 2. The actual cost function is denoted by J_μ^c. Assume that all trajectories that start off in X_0 belong to a bounded region $X_{inv} \supset X_0$, which is a subset of a bounded region X of the form

$$X = [x_1^l, x_1^h] \times [x_2^l, x_2^h] \times \cdots \times [x_n^l, x_n^h], \tag{6.3}$$

where $x_i^l < 0 < x_i^h$ for all $i = 1, \ldots n$. In other words, X is a rectangle in $\mathbf{R}^n$; the i-th coordinate of each vertex of X is either x_i^l or x_i^h. Region X_0 represents the region for which it is desirable to design an improved policy. We then select X as a superset of X_{inv} in order to include the trajectories of the closed loop system under policy μ which start off in X_0. For each coordinate i, we select a set $P_i = \{x_1^i = x_l^i, x_2^i, \ldots, x_{M_i}^i = x_h^i\}$ of M_i scalar reals, such that $x_1^i = x_l^i < x_2^i, < \ldots, < x_{M_i}^i = x_h^i$. The grid is defined as *the set of points of* $\mathbf{R}^n$ *whose i-th coordinate belongs to* P_i*, for each* i. An example of regions X_0, X_{inv}, X in $\mathbf{R}^2$ and the grid is shown in Figure 6.1.

6.1.2 Computing the Directional Derivative of the Cost Function

Recall from Section 2.2 that the directional derivative of the actual cost function J_μ^c along a direction $g(x_0)$ at a point x_0 is given by the limit $\lim_{t\to\infty} J_m(x_0, t)$, where J_m is defined in (2.23). Recall from (2.25) that J_m obeys the differential equation

$$\frac{\partial J_m(x_0,t)}{\partial t} = [2x_{x_0}^T(t,0)Q + \mu(x_{x_0}(t,0))^T R \frac{\partial \mu(x_{x_0}(t,0))}{\partial x_{x_0}}] \cdot x_{x_0\delta}(t,0), \quad J_m(x_0,0) = 0, \tag{6.4}$$

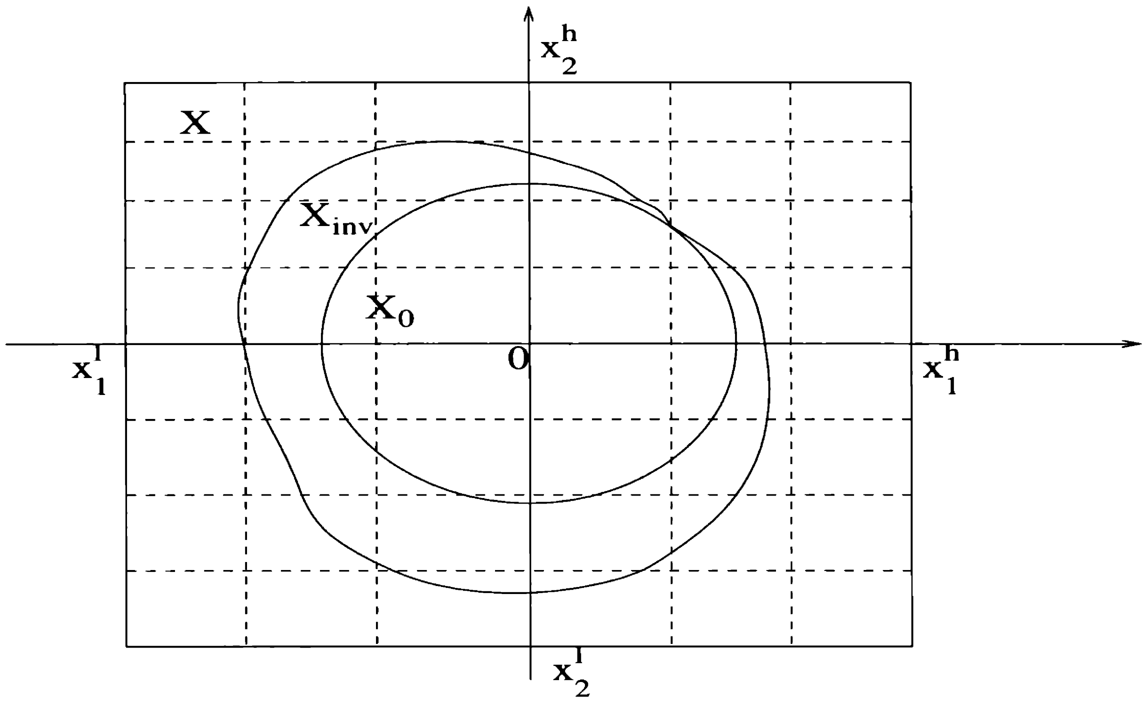

Figure 6.1: An example of the regions X_0, X_{inv}, X and the grid in $\mathbf{R}^2$.

where $x_{x_0}(t,0)$ is defined in (2.24) as the solution of the system's dynamics (6.2) with initial condition $x_{x_0}(t=0,0)=x_0$, and $x_{x_0\delta}(t,0)$ is given as the solution of a system of differential equations (2.15), with initial condition $x_{x_0\delta}(0,0)=g(x_0)$. The differential equations (6.4), (6.2), (2.15) form a system of differential equations, whose solution converges as $t \to \infty$. Under initial conditions as described above, the limit of equation (6.4) as $t \to \infty$ converges to $L_g J_\mu^c(x_0)$. Therefore, we integrate this system of differential equations numerically in order to compute $L_g J_\mu^c(x_0)$, in the case that the input matrix is a column vector g (scalar input case). In the case that the input matrix G is a $n \times m$ matrix (the input belongs to $\mathbf{R}^m$ and $G(x_0)=[g_1(x_0),\ldots,g_m(x_0)]$), then each directional derivative $L_{g_i} J_\mu^c(x_0)$, for $i=1,\ldots,m$ is computed via integration of the system of differential equations with the appropriate initial conditions ($J_m(x_0,0)=0$, $x_{x_0}(t=0,0)=x_0$, $x_{x_0\delta}(0,0)=g_i(x_0)$). Then, $L_G J_\mu^c(x_0)=[L_{g_1}J_\mu^c(x_0),\cdots,L_{g_m}J_\mu^c(x_0)]^T$ by definition (1.6). The strategy for numerical integration could be as simple as an Euler scheme with a small interval δ, as described in Chapter 1. In the next section we propose an affine interpolation

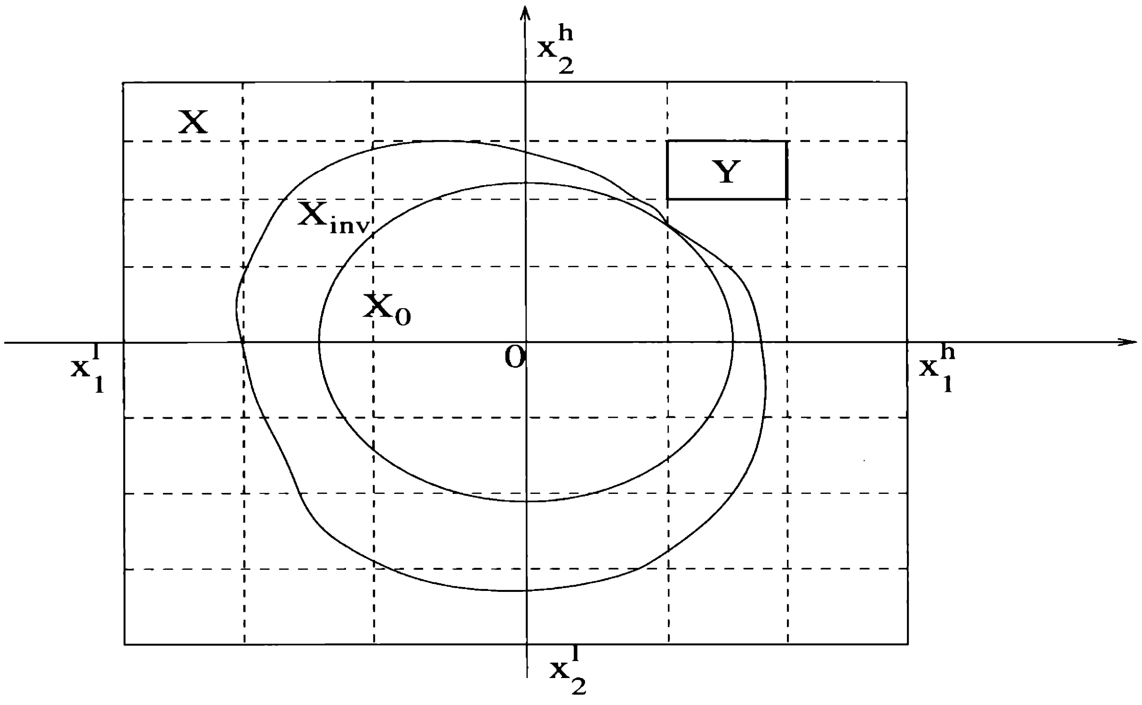

Figure 6.2: An example of an elementary subrectangle Y in $\mathbf{R}^2$.

scheme $\tilde{L}_{Y,i}(x)$ as an approximation of $L_{g_i(x)}J_\mu^c(x)$ in Y, for $i = 1, \ldots, m$.

6.1.3 Implementation of Approximate Policy Iteration

We numerically compute the directional derivative $L_G J_\mu^c$ at all vertices of the grid. We refer to any subrectangle of the grid which does not include any points of the grid other than its own vertices as an *elementary subrectangle.* Each elementary subrectangle has 2^n vertices, where n is the dimension of the state space. Consider an elementary subrectangle, which we denote by Y throughout the rest of the chapter for notational convenience. See Figure 6.2. Consider some x that belongs to Y, and we select $\tilde{L}_{Y,i}$ in the form of an affine function $x^T c_{Y,i}^\mu + \alpha_{Y,i}^\mu$ of x for $i = 1, \ldots, m$, where $c_{Y,i}^\mu$ is a column vector of dimension n and $\alpha_{Y,i}^\mu$ a scalar. The affine functional is selected such that, for each i, $x^T c_{Y,i}^\mu + \alpha_{Y,i}^\mu$ is an affine interpolation of the values of $L_{g_i} J_\mu^c$ at the vertices of Y. The interpolation scheme is such that the vertices that are closer to x have a higher contribution to the value of $x^T c_{Y,i}^\mu + \alpha_{Y,i}^\mu$. Such is the scheme used in Chapter 8 for control of the beam-and-ball system. A different affine interpolation possibility involves a least square fitting of an affine functional to the values at the vertices. Both these interpolation schemes are discontinuous at the facets of the rectangle. Remarks 1, 2 and 3 below include some important comments on the issue of interpolation. The approximation $\tilde{L}(x)$ is formed as the vector $[\tilde{L}_1, \ldots, \tilde{L}_m]^T$, and the updated controller μ' takes the following form in Y:

$$\mu'(x) = -\frac{1}{2}R^{-1}\tilde{L}_Y(x) = -\frac{1}{2}R^{-1}\left[x^T c^{\mu}_{Y,1} + \alpha^{\mu}_{Y,1}, \ldots, x^T c^{\mu}_{Y,m} + \alpha^{\mu}_{Y,m}\right]^T \qquad (6.5)$$

Remark 1:

We note that for $n = 1$ an affine averaging scheme with the property that $x^T c^{\mu}_{Y,i} + \alpha^{\mu}_{Y,i}$ is equal to the actual value of $L_{g_i} J^c_{\mu}$ at the 2 vertices of Y can be selected. However, this is not possible for any $n \geq 2$ and rectangular gridding. Indeed, if that was to be the case, then a system of 2^n equations (one for each vertex) with $n+1$ unknowns (the n elements of $c^{\mu}_{Y,i}$ plus $\alpha^{\mu}_{Y,i}$) would be required to have a solution. Any affine averaging scheme thus fails to generate the right value at the vertices. Nevertheless, it should accomplish to generate a good approximation of the actual function throughout Y. An example of such an affine representation is given in Chapter 8 and successfully applied to a beam-and-ball problem.

Remark 2:

Following up on Remark 1, let us note here that, given a subrectangle Y, we may adopt a strategy of computing the actual value of $L_{g_i} J^c_{\mu}$ only at a number of $n + 1$ vertices, and then determine an affine interpolation scheme based on those via solution of a system of $n + 1$ algebraic equations with $n + 1$ unknowns. In that case, we may be ignoring some potentially vital information, but we reduce the computational requirements of the scheme significantly, especially as the dimension of the state space is large, in which case computing $L_{g_i} J^c_{\mu}$ at $n + 1$ rather than at 2^n vertices might be of great benefit. Such a scheme remains to be tested in practice in order to evaluate its practicality for approximation. An alternative strategy, used widely in finite element methods [42], involves appropriately dividing the state space in simplices instead of rectangles, and it results in an affine approximation that takes the actual values at the vertices of the simplices. In $\mathbf{R}^2$, the simplex is a triangle. In $\mathbf{R}^3$ it is a tetrahedron. In any higher dimension, n, the corresponding simplex is the generalization of the triangle and the tetrahedron in $\mathbf{R}^n$. For rigorous definitions, see [45] or other texts in algebraic topology. The simplices are convex sets. Therefore, tests such as the one developed below in Section 6.2, for a rectangular gridding can be developed for a simplice gridding. Another alternative which should be explored, involves a polynomial interpolation of appropriate higher degree, depending on the dimension of the state space, such that the interpolation generates the right value at all vertices. We do not pursue this direction any further, but we recommended it for future research in Section 9.1. In this thesis, we restrict our attention to rectangles and affine interpolation approximation for simplicity and in order to demonstrate the main features of a grid-based approximation strategy.

Remark 3:

The piecewise affine interpolation schemes proposed above are discontinuous at the facets of the subrectangles in general, whereas the scheme proposed in Remark 2 is discontinuous. It is easy to show that such such piecewise affine schemes can be approximated very well by smooth functions. For an example with a function in one real variable, see Figure 6.3. In that case, the results of Chapter 2 go through without need for more elaborate mathematical arguments in order to account for nondifferentiability or discontinuity, while the estimation tests presented in Section 6.2 are still quite accurate since the piecewise affine function is approximated very well by its smooth counterpart. Therefore, in the next section we work with piecewise affine functions which enable us to obtain easily computable checks of stability of the iterate.

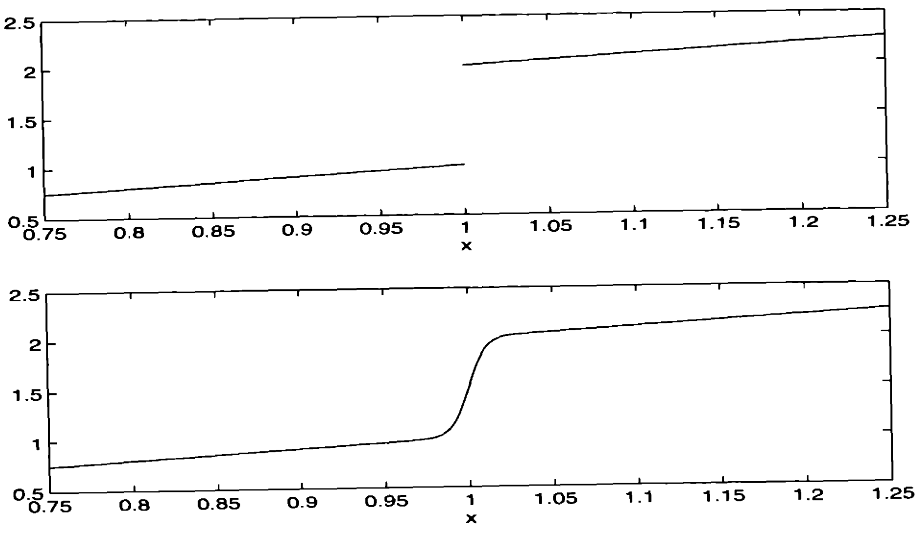

Figure 6.3: The top plot is the graph of a piecewise continuous function $y(x)$ given by $y(x) = x$ if $x \leq 1$ and $y(x) = x+1$ if $x > 1$. The bottom plot is the graph of the smooth function $y'(x) = \frac{1}{2}\{[1 - \tanh(100(x-1))]x + [1 + \tanh(100(x-1))](x+1)\}$, which is a good approximation of $y(x)$.

6.2 A Convex Optimization Test for Checking Asymptotic Stability of the Iterate

For simplicity of notation, we consider the case of a scalar input, that is, $m = 1$. Also for simplicity, we assume unity state and input weights in the cost function, $Q = I$ and $R = 1/2$. Then, $L_g J_\mu^c$ is approximated by $\tilde{L}_Y(x) = x^T c_Y^\mu + \alpha_Y^\mu$ in Y, and the updated policy in Y is given by

$$\mu'(x) = -\tilde{L}_Y(x) = -(x^T c_Y^\mu + \alpha_Y^\mu). \tag{6.6}$$

Recall the criterion for asymptotic stability of the iterate μ' (Proposition 2.3.1). We consider asymptotic rather than exponential stability for simplicity, since the right side of the criterion (2.29) is 0 for asymptotic stability. The closed loop system under

μ' is asymptotically stable if

$$-x^T x - \frac{1}{4}\mu(x)^2 - (\frac{1}{2}\mu(x) + L_g J_\mu^c(x))^2 + L_g J_\mu^c(x)(L_g J_\mu^c(x) - \tilde{L}_Y(x)) < 0, \qquad (6.7)$$

for all $x \in$ Y, and all elementary subrectangles Y.

Assume that the policy μ is also of the form (6.6), that is, it is obtained via approximate policy iteration and gridding. Assume that the grid used for obtaining μ is the same as that used for approximating $L_g J_\mu^c$ in order to derive μ'. In other words, μ is assumed to be affine in every elementary subrectangle Y. Let μ have the form:

$$\mu(x) = -(x^T c_Y + \alpha_Y), \quad x \in \mathrm{Y} \qquad (6.8)$$

Let us focus on an elementary subrectangle Y. Our objective in this section is to develop an algorithm for computing bounds on the difference between $L_g J_\mu^c(x)$ and $\tilde{L}_Y(x)$, for $x \in$Y such that (6.7) is satisfied throughout Y. Such bounds allow the designer to *estimate* the soundness of $\tilde{L}_Y$ as an approximator of $L_g J_\mu^c$ in Y, by comparing them to samples of the actual error at points in Y. Each sample can be obtained by direct computation of the $L_g J_\mu^c$ as described in Section 6.1.2.

Development of the Algorithm: We now develop the algorithm for computing the bounds in steps. We express the directional derivative of the actual cost function, $L_g J_\mu^c(x)$ as the product of a scalar $\epsilon(x)$ and the approximation $\tilde{L}_Y(x)$:

$$L_g J_\mu^c(x) = \epsilon(x)\tilde{L}_Y(x), \quad x \in \mathrm{Y} \qquad (6.9)$$

Then, the last of the terms appearing in the condition for asymptotic stability is written as

$$L_g J_\mu^c(x)(L_g J_\mu^c(x) - \tilde{L}_Y(x)) = \tilde{L}_Y(x)^2\epsilon(x)(\epsilon(x) - 1) \qquad (6.10)$$

Consider the case when $\epsilon(x) \in [0,1]$. Then, $\epsilon(x)(\epsilon(x)-1) \leq 0$ and the condition is automatically satisfied. Indeed, $\epsilon(x) \in [0,1]$ is the case when $\tilde{L}_Y(x)$ has the same sign as and larger absolute value than $L_g J_\mu^c(x)$. On the contrary, if $\epsilon(x) < 0$ or $\epsilon(x) > 1$, it has to be checked whether the expression at the right side of (6.10) is negative.

In the sequel, we assume that for all $x \in$Y, $\epsilon(x) \in [\epsilon_n, 0)$, for some scalar $\epsilon_n < 0$. We then develop a test such that if it succeeds, then it is guaranteed that the asymptotic stability test for $x \in$Y is satisfied. If the test fails, we conduct a binary search in $\epsilon_n < 0$ so as to find the smallest negative ϵ_n such that the test succeeds. Similarly, we assume that for all $x \in$Y, $\epsilon(x) \in (1, \epsilon_p]$, for some scalar $\epsilon_p > 1$. We repeat the same procedure and find the largest value of ϵ_p such that the test succeeds. Clearly, the case where $\epsilon(x) \in [\epsilon_n, 0)$ for some $x \in$Y, $\epsilon(x) \in [0,1]$ for some $x \in$Y and $\epsilon(x) \in (1, \epsilon_p]$ in the remainder of Y is covered by the combination of the two tests above. Thereby, we establish bounds on $\epsilon(x)$ such that (6.9) is satisfied in Y.

Step 1:

Assume that for all $x \in$Y, $\epsilon(x) \in [\epsilon_n, 0)$, for some scalar $\epsilon_n < 0$. Condition (6.7) is written as follows:

$$-x^T x - \frac{1}{4}\mu(x)^2 - (\frac{1}{2}\mu(x) + L_g J_\mu^c(x))^2 + L_g J_\mu^c(x)(L_g J_\mu^c(x) - \tilde{L}_Y(x)) < 0. \quad (6.11)$$

After simple algebraic manipulations, the terms that are quadratic in $L_g J_\mu^c(x)$ cancel out, and the condition becomes:

$$-x^T x - \frac{1}{2}\mu(x)^2 - L_g J_\mu^c(x)\mu(x) - L_g J_\mu^c(x)\tilde{L}_Y(x) < 0, \quad (6.12)$$

for all $x \in$ Y, which, in conjuction with (6.6), (6.8) and (6.10), becomes:

$$x^T \left[-I - \frac{1}{2}c_Y c_Y^T + \epsilon(x)\left(c_Y^\mu c_Y^T - c_Y^{\mu T}\right)\right] x$$

$$+x^T\left[-\alpha_Y c_Y + \epsilon(x)\left(\alpha_Y c_Y^\mu + \alpha_Y^\mu c_Y - 2\alpha_Y^\mu c_Y^\mu\right)\right] \\ + \left[-\frac{1}{2}\alpha_Y^2 + \epsilon(x)\left(\alpha_Y^\mu \alpha_Y - \alpha_Y^{\mu 2}\right)\right] < 0, \quad \text{for all} \quad x \in \mathrm{Y}. \tag{6.13}$$

We introduce some shorthand notation by defining the matrices

$$A_2(\epsilon) \triangleq \left[-I - \frac{1}{2}c_Y c_Y^T + \epsilon\left(c_Y^\mu c_Y^T - c_Y^{\mu T}\right)\right]$$
$$A_1(\epsilon) \triangleq \left[-\alpha_Y c_Y + \epsilon\left(\alpha_Y c_Y^\mu + \alpha_Y^\mu c_Y - 2\alpha_Y^\mu c_Y^\mu\right)\right]$$
$$A_0(\epsilon) \triangleq \left[-\frac{1}{2}\alpha_Y^2 + \epsilon\left(\alpha_Y^\mu \alpha_Y - \alpha_Y^{\mu 2}\right)\right].$$

We then pose the following optimization problem (P):

Compute ϵ_{low} defined as the infimum over all $\epsilon_n < 0$ such that

$$x^T A_2(\epsilon_n)x + x^T A_1(\epsilon_n) + A_0(\epsilon_n) < 0, \quad \text{for all } x \in \mathrm{Y} \tag{6.14}$$

and ϵ_{up} defined as the supremum over all $\epsilon_p > 1$ such that

$$x^T A_2(\epsilon_p)x + x^T A_1(\epsilon_p) + A_0(\epsilon_p) < 0, \quad \text{for all } x \in \mathrm{Y} \tag{6.15}$$

Then, clearly, condition (6.13) holds for every x in Y if $\epsilon(x) \in (\epsilon_{low}, \epsilon_{up})$ for every x in Y. Therefore, ϵ_{low} and ϵ_{up} constitute the bounds we are looking for. As mentioned above, the optimization problem (P) would be solvable via binary search over $\epsilon_n < 0$ and $\epsilon_p > 1$ if the feasibility conditions (6.14) and (6.15) were easy to check for a fixed value of ϵ_n and ϵ_p, respectively. However, they are not easy to check because the functionals on the left side of conditions (6.14) and (6.15) are nonconvex in general. In the next step, we introduce a modification of the feasibility problem (6.14) or (6.15) such that if the modified problem is feasible, then the original problem (6.14) or (6.15) respectively is feasible as well.

Step 2:

We consider the feasibility problem (6.14). The same arguments follow similarly for problem (6.15). It is possible to find a matrix $T_Y \in \mathbf{R}^{n\times n}$ and a vector $b_Y \in \mathbf{R}^n$ such that the affine coordinate transformation $w = T_Y x + b_Y$ maps the elementary subrectangle Y to a cube W defined as

$$W = \Big\{ w = [w_1, \ldots, w_n]^T \in \mathbf{R}^{n\times n} \quad \text{such that} \quad |w_i| \le 1, \quad i = 1, \ldots n \Big\} \tag{6.16}$$

The inequality $x^T A_2(\epsilon_n) x + x^T A_1(\epsilon_n) + A_0(\epsilon_n) < 0$ takes the form $w^T P_2(\epsilon_n) w + w^T P_1(\epsilon_n) + P_0(\epsilon_n) < 0$ in the new coordinate system, where the matrix $P_2(\epsilon_n)$, the vector $P_1(\epsilon_n)$ and the scalar $P_0(\epsilon_n)$ can be computed easily.

We now introduce yet another set of coordinates $z \in \mathbf{R}^n$ and $y \in \mathbf{R}$ and we let

$$w = \frac{z}{y}.$$

It follows that the inequality $w^T P_2(\epsilon_n) w + w^T P_1(\epsilon_n) + P_0(\epsilon_n) < 0$ is now written as

$$z^T P_2(\epsilon_n) z + z^T P_1(\epsilon_n) y + y^2 P_0(\epsilon_n) < 0$$

or, equivalently, in matrix form

$$\begin{bmatrix} z^T & y \end{bmatrix} \begin{bmatrix} P_2(\epsilon_n) & \frac{1}{2} P_1(\epsilon_n) \\ \frac{1}{2} P_1(\epsilon_n)^T & P_0(\epsilon_n) \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} < 0. \tag{6.17}$$

An equivalent form of the constraint $|w_i| \le 1$ is $w_i^2 \le 1$ which in the coordinates z, y

takes the form $z_i^2 \leq y^2$, for $i = 1, \dots n$. The latter takes the matrix form

$$\begin{bmatrix} z^T & y \end{bmatrix} \begin{bmatrix} K_i & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} \leq 0, \tag{6.18}$$

where K_i is the $(n \times n)$-matrix with all elements 0 except the i-th element of its diagonal, which is equal to 1. With all the above changes of coordinates, the original feasibility problem (6.14) is transformed into the problem of feasibility of (6.17) under constraints (6.18) for $i = 1, \dots, n$. The left sides of these inequalities are nonconvex functions of $[z^T \quad y]^T$ in general. However, a Lagrangean multiplier method called the S-procedure [81, 16] can be used to obtain a related convex feasibility problem. The S-procedure amounts to posing the following problem:

Determine if there exist positive scalars $\tau_1, \dots, \tau_n$ such that

$$\begin{bmatrix} P_2(\epsilon_n) & \frac{1}{2}P_1(\epsilon_n) \\ \frac{1}{2}P_1(\epsilon_n)^T & P_0(\epsilon_n) \end{bmatrix} - \tau_1 \begin{bmatrix} K_1 & 0 \\ 0 & -1 \end{bmatrix} + \cdots - \tau_n \begin{bmatrix} K_n & 0 \\ 0 & -1 \end{bmatrix} < 0. \tag{6.19}$$

It can be easily verified that if (6.19) is satisfied for some positive scalars $\tau_1, \dots, \tau_n$, then the problem (6.17) under constraints (6.18) is feasible, and, consequently, the original problem (6.14) is feasible. The problem (6.19) is a convex feasibility problem, and can be solved via efficient computational methods [48]. Let us note here that whereas feasibility of problem (6.19) implies feasibility of the original problem (6.14), the converse does not hold in general. Therefore, this approach may be conservative in the sense that the smallest $\epsilon_n < 0$ such that (6.19) is feasible may be larger than ϵ_{low}. Similar arguments hold for the feasibility problem (6.15). However, computational experience shows that the conservatism may be quite low in many practical cases. An approach to estimating the conservatism of the S-procedure and a thorough computational study, in a different context, is given in [15].

A similar test can be developed along the lines of Steps 1 and 2 for computing the supremum over $\epsilon_p > 1$ such that the criterion for asymptotic stability holds (with some conservatism due to the S-procedure).

To summarize the overall procedure, for each subrectangle Y, we conduct a binary search over $\epsilon_n < 0$ to determine (to a specified degree of accuracy) the infimum over $\epsilon_n < 0$ such that (6.19) holds. Let $\bar{\epsilon}_n$ be that number. Along the same lines, we determine a number $\bar{\epsilon}_p > 1$, such that if

$$\epsilon(x) \in \quad [\bar{\epsilon}_n, \bar{\epsilon}_p], \qquad \text{for all } x \in Y, \tag{6.20}$$

then the criterion for asymptotic stability of μ' is satisfied. We then collect some samples of $\epsilon(x)$ randomly throughout Y via simulation. We repeat the same process for every elementary subrectangle Y. If any of these samples in any subrectangle does not satisfy (6.20), then we refine the grid and repeat the same procedure. If, on the other hand, all the samples of $\epsilon(x)$ in all subrectangles satisfy (6.20), there is still no guarantee that the criterion is satisfied everywhere. Nevertheless, this is strongly suggested. It is finally up to the designer to accept or refine the current grid. The decision can be based on factors like how large is the number of samples of $\epsilon(x)$ that were obtained, and how close are these samples to the boundaries $\bar{\epsilon}_n$ and $\bar{\epsilon}_p$ of the allowed interval. Based on these factors, the designer may decide to refine the grid at certain parts of the state space and keep it as it is (or even make it more coarse) in other areas.

Finally, in Section 9.1 we discuss the possibility of a convex feasibility test for determining bounds on $\epsilon(x)$ throughout the state space. Such a test might be possible by virtue of the fact that $L_g J_\mu^c$ is given as the solution of a system of *linear parameter varying* differential equations (see Section 2.2). Such a test, in conjuction with the test of this section, would form a (possibly conservative) entirely systematic way for checking whether a specific grid-based approximation satisfies the criterion for asymptotic stability. This topic is recommended for future research.

Chapter 7

Missile Autopilot Design

In this chapter, we use approximate policy iteration to design a nonlinear controller for a missile. We follow the procedure described in Chapter 1. The initial controller is a gain-scheduled controller given by Shamma and Cloutier in [59]. The ideas presented in Chapter 5 are used in selection of an appropriate cost function approximation architecture. The objective is controller design such that the closed loop system achieves perfect normal acceleration tracking of varying size step input commands. The punchline of the chapter is depicted in Figure 7.2. The figure demonstrates that approximate policy iteration results in about 8-fold reduction in the cost of the closed loop systems under the generated controllers between iterations 2 and 22, whereas a comparison to Shamma and Cloutier's controller is not applicable because we augment the system dynamics we explain in the sequel.

7.1 Problem Formulation

7.1.1 Missile Dynamics

The missile dynamics considered here are taken from [52]. They are representive of a missile traveling at Mach 3 at an altitude of 20,000 ft. It is a 2-dimensional dynamic system, with states α (angle of attack, measured in degrees) and q (pitch rate, measured in deg/sec). The dynamic equations are as follows:

$$\dot{\alpha} = f\frac{g\cos(\alpha/f)}{WV}Z + q \tag{7.1}$$

$$\dot{q} = fm/I_{yy}, \tag{7.2}$$

where $d = 0.75$ ft. is the reference diameter, $f = 180/\pi$ is the radians-to-degrees conversion factor, $g = 32.2$ ft/s^2 is the acceleration of gravity, $I_{yy} = 182.5$ slug-ft^2 is the pitch moment of inertia, $m = C_m QSd$ is the pitch moment in ft-lb, $Q = 6132.8$ lb-ft^2 is the dynamic pressure, $S = 0.44$ ft^2 is the reference area, $V = 3109.3$ ft/s is the speed, $W = 450$ lb is the weight, and $Z = C_Z QS$ is the normal force in lb. The normal force and pitch moment aerodynamic coefficients are approximated by

$$C_Z = 0.000215\alpha^3 - 0.0195\alpha|\alpha| + 0.051\alpha - 0.034\delta$$
$$C_m = 0.000103\alpha^3 - 0.00945\alpha|\alpha| - 0.170\alpha - 0.206\delta,$$

where by δ we denote the fin deflection, which is the control input of the missile. These approximations are accurate for α in the range of ± 20 deg.

The objective of the autopilot is to control the normal acceleration

$$\eta_Z = Z/W, \tag{7.3}$$

which is measured in g. In modern air-to-air or surface-to-air missiles, an onboard guidance system (e.g. of the proportional navigation or line-of-sight type) provides appropriate normal acceleration commands by using, for example, the reflected radio frequency energy resulting from an illuminating radar sitting on the ground or carried onboard the missile for course adjustment [12]. The guidance system updates the normal acceleration command on line so as to track the target, whereas the autopilot aims at achieving command following. The autopilot performance objective is to track normal acceleration step commands of size in the range ±20g, with a steady state accuracy of less than 0.5% and a time constant of 0.2sec.

In order to simplify notation, we denote the two states of the plant by x_1 and x_2 (angle of attack and pitch rate, respectively) and the control input (fin deflection) by u. The plant dynamics then take the form

$$\dot{x}_1 = \cos(\frac{x_1}{a_1})(a_2 x_1^3 + a_3 x_1|x_1| + a_4 x_1) + x_2 + \cos(\frac{x_1}{a_1})a_5 u \tag{7.4}$$

$$\dot{x}_2 = a_6 x_1^3 + a_7 x_1|x_1| + a_8 x_1 + a_9 u, \tag{7.5}$$

where $a_1, \cdots, a_9$, are constants which can be inferred from equations (7.1), (7.2) and the definitions of Z and m. The output (normal acceleration) is denoted by y, and it is given by

$$y = a_{10}(a_2 x_1^3 + a_3 x_1|x_1| + a_4 x_1) + (a_{10} a_5)u \tag{7.6}$$

where a_{10} is a constant which can be inferred from the definition of the normal acceleration (7.3) and the definitions of Z, W and a_2, a_3, a_4, a_5.

For simulation purposes, we introduce the following discrete time representation of the plant dynamics, as discussed in Chapter 1:

$$x_1(t+1) = \quad x_1(t) + \delta \left\{ \cos(\tfrac{x_1(t)}{a_1})(a_2 x_1(t)^3 + a_3 x_1(t)|x_1(t)| + a_4 x_1(t)) \right.$$

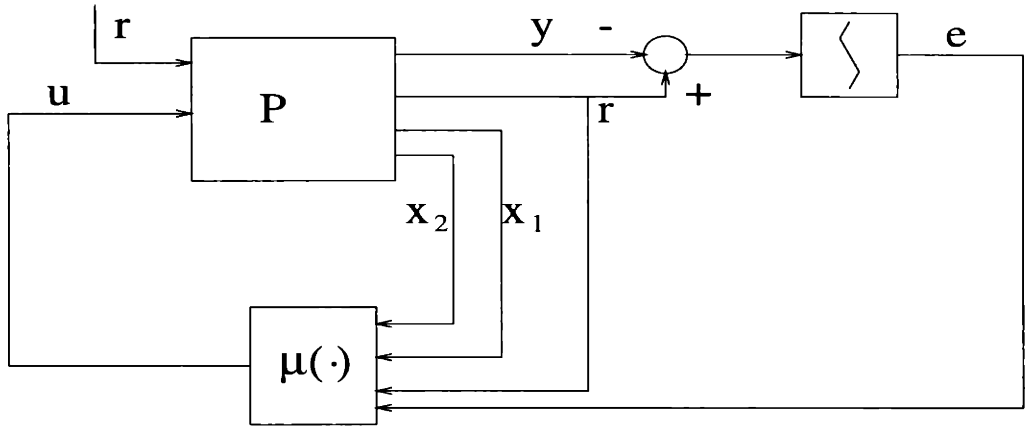

Figure 7.1: The augmented system: the plant P denotes the missile

$$+x_2(t) + \cos(\tfrac{x_1(t)}{a_1})a_5u(t)\Big\} \qquad (7.7)$$

$$x_2(t+1) = \quad x_2(t) + \delta\{a_6x_1(t)^3 + a_7x_1(t)|x_1(t)| + a_8x_1(t) + a_9u(t)\} \qquad (7.8)$$

$$y(t) = \quad a_{10}(a_2x_1(t)^3 + a_3x_1(t)|x_1(t)| + a_4x_1(t)) + (a_{10}a_5)u(t) \qquad (7.9)$$

7.1.2 State Augmentation

To ensure perfect tracking, we introduce an integrator in the loop, as shown in Figure 7.1. The integrator is implemented as an accumulator in discrete time, and it constitutes an extra dynamic component whose state is a scalar e evolving in time as:

$$e(t+1) = e(t) + \delta\left[r(t) - y(t)\right], \qquad (7.10)$$

where $r(t)$ is the step command. It can be clearly seen that unless the tracking error $r(t) - y(t)$ is zero, the accumulator is not at equilibrium. To account for the varying size of the step command, we view the step command $r(t)$ as an extra state variable, with dynamics

$$r(t+1) = r(t), \qquad r(0) = r_0, \tag{7.11}$$

where r_0 is the size of the command. By viewing $r(t)$ as an extra state of the system, we ensure that the controllers obtained via approximate policy iteration satisfy the command following specification regardless of the step size. The resulting augmented dynamic system is shown in Figure 7.1, and its dynamics are given by

$$\begin{aligned}
x_1(t+1) &= x_1(t) + \delta\Big\{\cos(\tfrac{x_1(t)}{a_1})(a_2x_1(t)^3 + a_3x_1(t)|x_1(t)| + a_4x_1(t)) \\
&\qquad + x_2(t) + \cos(\tfrac{x_1(t)}{a_1})a_5u(t)\Big\} \\
x_2(t+1) &= x_2(t) + \delta\{a_6x_1(t)^3 + a_7x_1(t)|x_1(t)| + a_8x_1(t) + a_9u(t)\} \\
e(t+1) &= e(t) + \delta\,[r(t) - y(t)]\,, \\
r(t+1) &= r(t) \\
y(t) &= a_{10}(a_2x_1(t)^3 + a_3x_1(t)|x_1(t)| + a_4x_1(t)) + (a_{10}a_5)u(t).
\end{aligned} \tag{7.12}$$

7.1.3 Optimal Control Problem

If the size of the step command is r_0, that is, $r(t) = r_0$ for all $t \geq 0$, then the objective is that the output asymptotically converges to r_0 as $t \to \infty$. From the dynamics of the system, it can be shown that, for every r_0, there exist functions $x_{1,ss}(r_0)$, $x_{2,ss}(r_0)$ and $u_{ss}(r_0)$ such that, as $t \to \infty$,

$$\begin{gathered}
y(t) \to r_0 \quad \text{if and only if} \\
x_1(t) \to x_{1,ss}(r_0) \quad \text{and} \quad x_2(t) \to x_{2,ss}(r_0) \quad \text{and} \quad u(t) \to u_{ss}(r_0).
\end{gathered} \tag{7.13}$$

Of those, u_{ss} is obtained as a function of $x_{1,ss}$ (angle of attack) by setting the left hand side of (7.5) to 0 and replacing x_1 and u by $x_{1,ss}$ and u_{ss}, respectively. Then, the expression of u_{ss} as a function of $x_{1,ss}$ can be replaced in the output equation (7.6), and y by r_0 to obtain an algebraic expression whose solution gives $x_{1,ss}$ as a function of r_0. This expression has multiple roots. However, plant specific dynamic considerations can assist us to select the right one, as follows. In specific, a positive (negative), constant normal acceleration y can only be generated by a negative (positive), constant angle of attack x_1 (see ([12], page 240). Let us consider the case where r_0 is positive. The algebraic expression mentioned above has two negative roots. However, in the output range of interest, that is, $[0, 20\text{g}]$ of r_0, only one of the two negative roots is in the $[-20\text{deg}, 0]$ range where the normal force and pitch moment aerodynamic coefficient approximations hold (see Section 7.1.1). In fact, the root that is discarded takes values in the below -100deg range. Finally, exactly analogous arguments hold in the case where r_0 is negative. The remaining steady state value, $x_{2,ss}$, can be easily obtained via (7.4).

The above discussion reveals that an algebraic expression of degree 3 has to be solved on-line in order to obtain the values $x_{1,ss}(r_0)$ $x_{2,ss}(r_0)$ and $u_{ss}(r_0)$. This may be difficult to implement in practice. Instead, we can tabulate the solutions of the expression (computed off-line) and use a look-up table in order to use those values on-line.

Furthermore, since the controller that we use (to be described shortly) uses feedback from the integrator state, there exists a function $e_{ss,\mu}(r_0)$ which gives the steady state value of the integrator state for a given controller such that $y(t) \to r_0$ is equivalent to (7.13) and $e(t) \to e_{ss,\mu}(r_0)$ for the closed loop system under controller μ. This value depends on the particular controller, the command r_0, and the steady state values $x_{1,ss}(r_0)$, $x_{2,ss}(r_0)$, $u_{ss}(r_0)$. In addition, it would be desirable to impose an integrator state initial condition $e(0)$ which is close to e_{ss}, so as to avoid undesirable output transients caused by the long integrator state transients.

For a closed loop under a controller μ, we define the following cost function:

$$\begin{aligned} J_\mu(x_0) = & \textstyle\sum_{t=0}^{\infty} \delta \left\{ (x_1(t) - x_{1,ss}(r_0))^2 + (x_2(t) - x_{2,ss}(r_0))^2 \right. \\ & \left. +20(e(t) - e_{ss,\mu}(r_0))^2 + (\mu(x(t)) - u_{ss}(r_0))^2 \right\}, \end{aligned} \tag{7.14}$$

where the trajectory $[x_1(t), x_2(t), e(t), r(t)]^T$ results from initial condition $x_0 = [x_{1,0}, x_{2,0}, e_0, r_0]^T$. The relatively high weight on the integrator state is an indirect attempt to increase the damping of the resulting closed loop systems under the designed controllers. Other than that, there is no other considerations involved in selecting the weights of the cost function. The main purpose of this application is to demonstrate the possibility of approximate policy iteration as a tool of improving on a cost function. No cost function adjustment has taken place in order to improve the transient responses of the obtained designs. In a sense, we do not focus as much on meeting the particular transient response requirements posed in Section 7.1.1 as on improving on the cost function (7.14). Nevertheless, a comparison between the controllers obtained and a controller specifically designed to address the missile transient response specifications follows at the end of the chapter.

Since the difference $(e(t) - e_{ss,\mu}(r_0))$ depends on the controller μ, we define the variable $\tilde{e}(t)$ as

$$\tilde{e}(t) \triangleq e(t) - e_{ss,\mu}(r_0), \tag{7.15}$$

which obeys (7.10), that is, the same dynamic equation as $e(t)$.

7.2 Iteration 1

In [59], Shamma and Cloutier develop a gain-scheduled controller for the missile control problem. Let us denote Shamma and Cloutier's controller by μ_0. This controller receives feedback from the angle of attack x_1, the pitch rate x_2, the normal acceleration y, and from the "dummy" reference command state variable r to generate the control input u. It is a dynamic controller, linear in x_2 and nonlinear in x_1 and r The state $e(t)$, that is introduced by us, is not part of the closed loop under μ_0. The term of the cost function that penalizes the integrator state, $20(e(t) - e_{ss,\mu}(r_0))^2$, is totally meaningless in Iteration 1, since $e_{ss,\mu}(r_0)$ is non-existent. In Iteration 1, instead of the term $20(e(t) - e_{ss,\mu}(r_0))^2$ we use the term $\frac{1}{2}(e(t) - e_f inal)^2$, where e_{final} is the *final value of the integrator state* at the time that the trajectory ends (1.0 sec, as described below). This replacement is a heuristic, which is motivated by the fact that the output tracking error of the closed loop system under μ_0 is small, and therefore the rate at which $e(t)$ grows is small as soon as steady state has been reached, but justified only by the fact that it works and provides an asymptotically stable controller μ_1 after one policy iteration. The weight factor of $\frac{1}{2}$ was also used for this term in Iteration 1, instead of 20, so as to reduce the weight of this heuristic term in the cost function.

The controller μ_0 does not achieve perfect tracking. At steady state, there is a small tracking error $r_0 - y(t)$, where r_0 is the size of the step command. Consequently, the cost function J_{μ_0} as defined in (7.14) for the system (7.12) under control μ_0, is not finite. However, we choose to apply approximate policy iteration to this system, using a simple heuristic. We simulate trajectories of the closed loop system under μ_0 until a fixed time $T = 1.0$ sec (the trajectories under μ_0 reach steady state by $T = 1.0$ sec) and we do approximate policy iteration on the basis of the cost accumulated between time 0 and 1.0 sec. From now on, this is what we denote by $J_{\mu_0}(x_0)$ (for any other policy μ, J_μ is still defined by the infinite sum (7.14)).

We compute via simulation the cost function J_{μ_0} at 400 sample points, which are chosen randomly and uniformly in the region:

$$X_0 = [-20, 20] \times [-40, 40] \times [e_{ss,\mu}(r_0) - 2.5, e_{ss,\mu}(r_0) + 2.5] \times [-20, 20]. \tag{7.16}$$

For example, the initial value of the pitch rate x_2 belongs to the interval $[-40, 40]$ for all simulated trajectories, and the size of the step command r_0 belongs to the interval $[-20, 20]$ for all simulated trajectories. Notice, that we evaluate the cost function at integrator state values ranging close to its steady state value, $e_{ss,\mu}(r_0)$. It turns out that this range is adequate for approximation architecture parameter tuning. It reflects our desire to impose integrator state initial conditions that are close to the steady state value so as to avoid possible large output transients triggered by large integrator state transients.

7.2.1 Approximation Architecture

We observe that the dynamics of the closed loop system under μ_0 are linear with respect to x_2 and e. Therefore, we try approximation architectures that are quadratic in x_2 and e. After choosing an architecture, we apply policy update. We keep changing the architecture, until obtaining an updated controller which performs satisfactorily. After some trial and error, we end up with the architecture:

$$\begin{aligned}\tilde{J}_{\mu_0}\left([x_1, x_2, e, r]^T\right) = \ & m_1 l(x)^2 + m_2 l(x)(x_1 - x_{1,ss}) + m_3 l(x)(x_2 - x_{2,ss}) + \\ & + m_4 l(x)e + m_5 l(x)r + m_6(x_1 - x_{1,ss})^2 + \\ & + m_7(x_2 - x_{2,ss})^2 + m_8 e^2 + m_9 r^2 + \\ & + m_{10}(x_1 - x_{1,ss})(x_2 - x_{2,ss}) + m_{11}(x_1 - x_{1,ss})e + \\ & + m_{12}(x_1 - x_{1,ss})r + m_{13}(x_2 - x_{2,ss})e + \\ & + m_{14}(x_2 - x_{2,ss})r + m_{15}er, \end{aligned} \tag{7.17}$$

where

$$l(x) \triangleq a_2(x_1 - x_{1,ss})^3 + a_3(x_1 - x_{1,ss})|x_1 - x_{1,ss}| + a_4(x_1 - x_{1,ss}),$$

and where by m_i we denote the tunable parameters of the architecture (the latter were denoted by r_i in Chapter 1, but we change that notation so as to avoid confusion with the reference command state $r(t)$).
The best values of m_i are determined on the basis of the 400 samples of J_{μ_0} by means of a least squares fit. The updated controller μ_1 is given by

$$\mu_1(x) = -L_g\tilde{J}(x) = -\nabla\tilde{J}(x) \cdot g(x), \tag{7.18}$$

where the input vector $g(x)$ is easily inferred from the dynamics (7.14), and $\nabla\tilde{J}(x)$ denotes the gradient of $\tilde{J}$ at x. The rightmost equality in (7.18) follows from differentiability of $\tilde{J}$. The updated controller is linear in x_2, e, as well as r.
The values of m_i, $i = 1, \ldots 15$ such that $\tilde{J}_{\mu_0}$ best approximates J_{μ_0} were computed as 0.8551, 3.4608, 0.0391, 0.2224, -0.5316, 5.7962, 0.1566, -0.7490, 1.0331, 0.7613, 0.2720, -3.8163, 0.0485, -0.5639, 0.3129, respectively. As explained above, these values are used to update the controller, that is, obtain μ_1.

7.3 Iterations 2 through 22

The closed loop system under controller μ_1 achieves perfect tracking of step commands. The cost function J_{μ_1} defined according to (7.14) is finite. Based on μ_1, we repeat policy iteration another 22 times. At each iteration, we use the same approximation architecture, and the cost function is sampled at 400 points that are randomly generated in the region X_0 defined by (7.16). These points are different

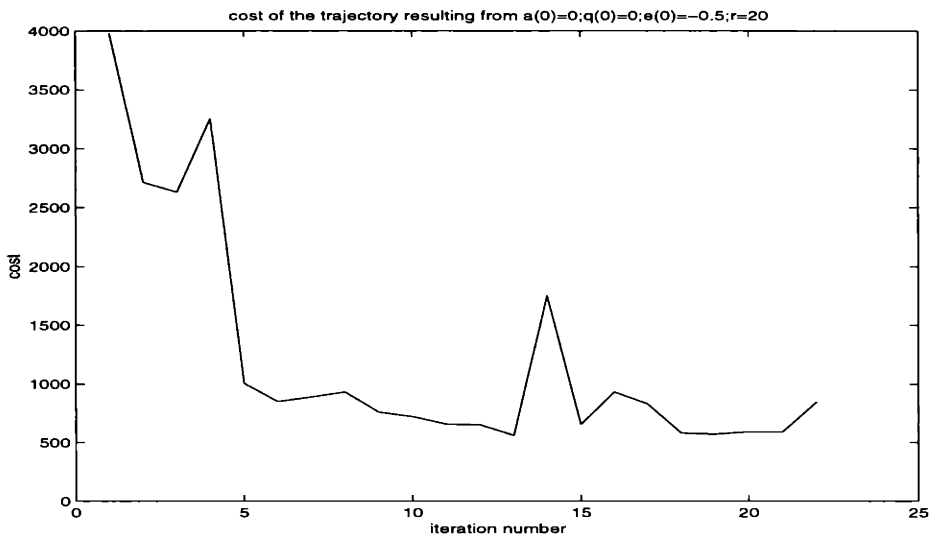

Figure 7.2: Decrease in value of the cost function evaluated at $x_1 = 0$ $x_2 = 0$, $\tilde{e} = -0.5$, $r = 20$ as the iteration number increases.

at each iteration. In Figure 7.2, we plot the value of the cost function at the point $x_1 = 0$ $x_2 = 0$, $\tilde{e} = -0.5$, $r = 20$. We see that the value of the cost function reduces by about a factor of 8. The decrease in cost is rapid over the first 6 iterations. After that, there is a decrease at a slower rate up to iteration 13. At iteration 14 we see a sharp increase, but the value decreases again at iteration 15. The oscillatory behavior observed after the iteration 6 is due to the fact that approximation is not perfect. Such an oscillatory behavior is also reported in the examples of Chapter 8 in [11]. Note that this behavior of the cost function as a function of iteration number shown in Figure 7.2 for a single value of the states, seems to be typical of all initial conditions that belong to X_0, as indicated by other simulations that we have performed.

In conclusion, the missile control application demonstrates the effectiveness of approximate policy iteration in producing controllers of improved performance. In this problem, the task of function approximation was made easier by taking into account the structure of the system dynamics.

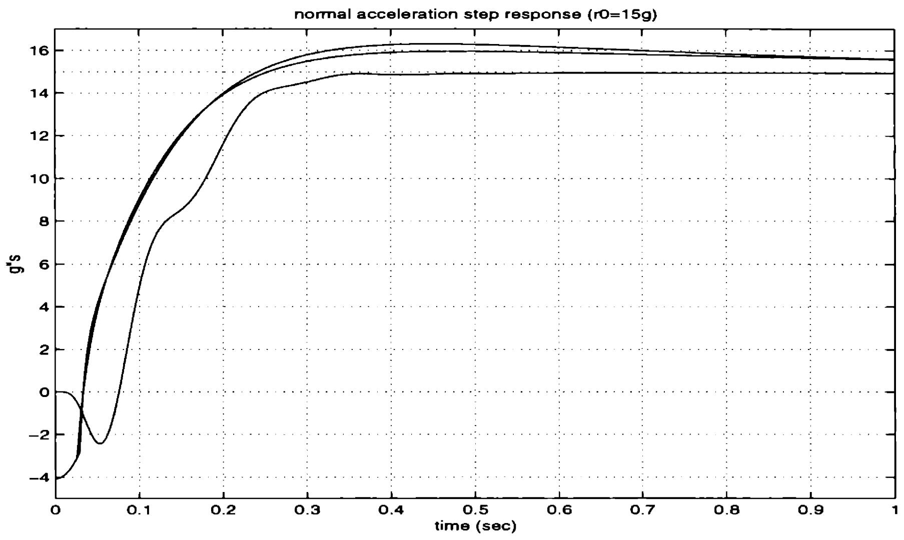

Figure 7.3: Normal acceleration response to a step command of size 15g of the closed loop systems under μ_0 (its response starts at 0 at time 0), under μ_8 (the response with the largest overshoot) and under μ_{16}

7.3.1 Sample Transient Responses

Although no effort was made to meet the step response performance criteria of 5% accuracy in less than 0.2sec, some of the generated controllers come very close to meeting those criteria. Note that even the initial controller, μ_0, of Shamma and Cloutier [59] fails to meet the 0.2sec criterion. As we see in the sequel, approximate policy iteration generates controllers under which the closed loop system is faster than under μ_0, but whose accuracy is worse. We also show representative transient response plots of some of the generated controllers which totally to meet the step response performance criteria.

We simulate the step response of the closed loop system under several controllers for step command size r_0 = 15g, and initial conditions $x_1(0) = 0$,$x_2(0) = 0$, and $\tilde{e}(0) = -0.5$. In these simulations, we include actuator saturation at $\pm$20deg. Actuator saturation were not part of the model used in approximate policy iteration to generate the controllers. It turns out that including has a minor effect on the closed loop

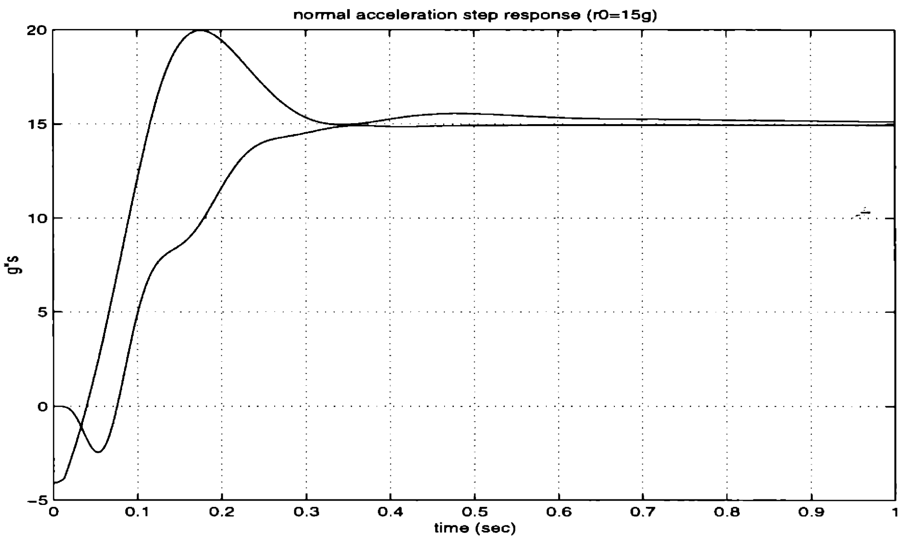

Figure 7.4: Normal acceleration response to a step command of size 15g of the closed loop systems under μ_0 (its response starts at 0 at time 0) and under μ_{14}.

responses, despite the fact that the unsaturated actuator commands can be quite large for a very small time at the beginning of the transients. Actuator dynamics also do not have but a minor effect, but this section's simulations do not include those.

Figure 7.3 shows the responses of the closed loop systems under μ_0, μ_8 and μ_{16}. Notice that the system under μ_0 does not satisfy the 0.2sec specification. Under either μ_8 or μ_{16}, the response is significantly faster. However, the 5% accuracy requirement is not satisfied; under μ_{16}, a 6.5% accuracy is achieved in 0.2sec, whereas under μ_8 a 10% accuracy in 0.18sec.

In Figure 7.4, we compare μ_0 to μ_{14}. The closed loop system under μ_{14} is about twice as fast as under μ_0, but it exhibits large overshoot.

Finally, in Figure 7.5 we show the responses under controllers μ_{11} and under μ_{20}. Both are slow and they overshoot. The presence of the integrator may guarantee that 100% accuracy is eventually achieved, but the transients are not good.

By comparing the responses of μ_8, μ_{11}, μ_{14}, μ_{16}, μ_{20}, we realize that they behave very differently, despite the fact that their costs are comparable (see Figure fig:costredu).

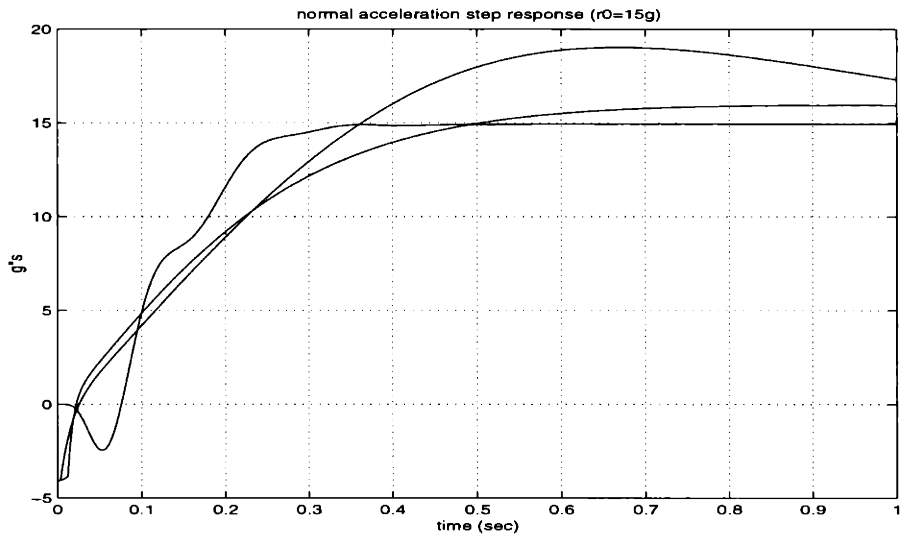

Figure 7.5: Normal acceleration response to a step command of size 15g of the closed loop systems under μ_0 (its response starts at 0 at time 0), under μ_{20} (the response with the largest overshoot) and under μ_{11}

However, we do not attempt to analyze this phenomenon any further.

The way to go about improving the transients of the closed loop systems under controllers generated by approximation policy iteration is by adjusting the cost function weights, like in the case of Linear Quadratic control. The sole objective of this chapter and the missile application is to demonstrate the success of approximate policy iteration in reducing the value of the cost function and show that the approach is an effective tool in an important real life control application.

Chapter 8

Beam-and-Ball Problem

The beam-and-ball problem has proved to be a challenging nonlinear control problem of stabilizing an unstable system. Thus, it is used as a benchmark problem in the field of nonlinear control. It remains a challenging benchmark for new control approaches, in spite of several recent interesting solutions [70, 55, 49]. We apply the approach presented in Chapter 4 for control of an unstable system, using the grid architecture proposed in Chapter 6.

The physical system is shown in Figure 8.1. The system description given below follows Section 5.4.3 in ([55]). The beam is assumed to be of infinite length for convenience. The beam is forced to rotate in a vertical plane by applying a torque at the center of rotation, and the ball is free to roll along the beam. In practice, the ball's motion is a combined roll and slip, for large values of the beam's angular acceleration, but here we assume that it only rolls. Let us denote by r the position of the ball on the beam, where $r = 0$ is the beam's center of rotation, by θ the angle of the beam, and by τ the control torque applied to the beam. Let the viscosity friction coefficient be $\beta = 0.1$, and the acceleration of gravity $g = 9.81$, and let the dynamic

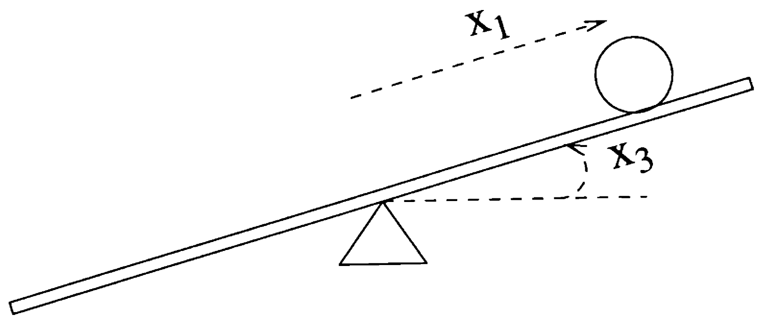

Figure 8.1: The beam and ball system: the torque is applied at the center of rotation. The beam's length is assumed infinite, although the beam of the figure is finite.

equations governing the system be

$$\begin{aligned} 0 &= \ddot{r} + g\sin(\theta) + \beta\dot{r} - r\dot{\theta}^2 \\ \tau &= (r^2+1)\ddot{\theta} + 2r\dot{r}\dot{\theta} + gr\cos(\theta) \end{aligned}$$

The state variables of the system are $x_1 = r$, $x_2 = \dot{r}$, $x_3 = \theta$ (in radians) and $x_4 = \dot{\theta}$ (in radians/sec). The state is the column vector $[x_1 \;\; x_2 \;\; x_3 \;\; x_4]^T$. The torque τ is the scalar control input u of the system. Then, the dynamics of the system can be written as :

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= -\beta x_2 - g\sin(x_3) + x_1 x_4^2 \\ \dot{x_3} &= x_4 \\ \dot{x_4} &= u \end{aligned}$$

We start with an initial policy $\mu_0(x) = 0$ for all x. The system under μ_0, that is, the open loop system, is unstable. The discrete time dynamics under the controller $\mu_0 = 0$ are given by

$$\begin{cases} x_1(t+1) &= x_1(t) + \delta x_2(t) \\ x_2(t+1) &= x_2(t) + \delta(-\beta x_2(t) - g\sin(x_3(t)) + x_1(t)x_4(t)^2) \\ x_3(t+1) &= x_3(t) + \delta x_4(t) \\ x_4(t+1) &= x_4(t) + \delta\mu_0(x(t)) = x_4(t) \end{cases} \tag{8.1}$$

For the unstable open loop system, we pose a discounted optimal control problem for a redefined dynamic system as in Chapter 4. The Training Region (TR) and the Region of Acceptable Operation (RAO) are defined later. Let us consider the stage cost $x^TQx + Ru^2$, where Q is a 4×4 symmetric positive definite matrix, and R is a positive definite scalar. Since the initial controller μ_0 is identically equal to 0 for every x, the value of R for the policy evaluation step of the 1st iteration is irrelevant. However, the controller μ_1 depends on R. We select $R = 0.1$. The weight matrix Q is a diagonal matrix $Q = \mathbf{diag}(Q_1, Q_2, Q_3, Q_4)$, and its elements are taken as $Q_1 = 4$, $Q_2 = 0.2$, $Q_3 = 40$, $Q_4 = 8$. The values of the cost function are chosen in a way that distributes the weight among the position of the ball, x_1, and the angle of the beam, x_3, roughly evenly. However, the angle rate x_4 is weighted about 10 times less, and the rate of change of the ball's position even less. Otherwise, if for example x_4 were heavily weighted, then the controller would tend to rotate the beam at low rates. Therefore, if the beam is tilted towards one side, it will keep tilting towards that direction for a long time, and during that time the ball will be rolling away from the center of rotation. In the case that x_2 is heavily weighted, then the controller will tend to keep the beam into the horizontal position, so that x_2 is small. But if the ball is not close to the center of rotation, it can only be forced quickly back to the

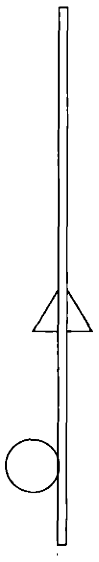

Figure 8.2: The initial condition from which the designed controllers are required to bring the system to equilibrium at $x = 0$.

center of rotation by tilting the beam at a large angle, so that the ball is accelerated by gravity towards the center of rotation. For the problem of stabilization, we select the Training Region (TR) as

$$\mathrm{TR} = \left\{ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} : \quad |x_1| \le 2, \;\; |x_2| \le 4, \;\; |x_3| \le 90\frac{\pi}{180}, \;\; |x_4| \le 90\frac{\pi}{180} \right\} \tag{8.2}$$

Our objective is to design a controller that brings the system from the initial position $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$ to the equilibrium position where all state variables are 0. The system configuration at the above initial condition is shown in Figure 8.2. Clearly, if the system is not controlled, the ball will roll-off to $-\infty$ under the influence of gravity. The Region of Acceptable Operation

(RAO) is selected as

$$\mathrm{RAO} = \left\{ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} : \quad [x_1 \ x_2 \ x_3 \ x_4] \begin{bmatrix} W_1 & 0 & 0 & 0 \\ 0 & W_2 & 0 & 0 \\ 0 & 0 & W_3 & 0 \\ 0 & 0 & 0 & W_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leq \ 150.0 \right\}, \tag{8.3}$$

where the square diagonal matrix $W = \mathbf{diag}(W_1, W_2, W_3, W_4)$ is selected equal to the state weight matrix of the cost function Q. Consequently, the cost function $J^d_{\mu_0}$ is constant over the boundary of RAO. We pose the problem of finding a policy that attains the minimum in

$$\begin{aligned} \arg\min_{\mu} \{ \qquad & \textstyle\sum_{t=0}^{N_\mu(x_0)} \alpha^t \cdot \delta \left(x^\mu_{x_0}(t)^T Q x^\mu_{x_0}(t) + \mu(x^\mu_{x_0}(t))^T R \mu(x^\mu_{x_0}(t)) \right) + \\ & + M \cdot \textstyle\sum_{t=N_\mu(x_0)+1}^{\infty} \alpha^t \cdot \delta \left(x^\mu_{x_0}(t)^T Q x^\mu_{x_0}(t) + \mu(x^\mu_{x_0}(t))^T R \mu(x^\mu_{x_0}(t)) \right) \Big\}, \\ & \text{for every } x_0 \in \mathrm{TR}, \end{aligned} \tag{8.4}$$

which corresponds to a integral quadratic cost for the continuous time problem, as discussed in Section 4.4. Recall from Chapter 4 that by $N_\mu(x_0)$ we denote the instant when the trajectory starting off at x_0 at time 0 leaves RAO. For all times after $N_\mu(x_0)$, the value of the state does not change in accordance with the redefined dynamics, as described in Chapter 4. The discount factor is chosen as $\alpha = 0.995$, and the unacceptable operation weight factor is given the value $M = 20$. The sampling interval δ is chosen at 0.0001sec. Finally, the "radius" 150 of the RAO (8.3) has been chosen slightly smaller than twice the maximum of the distance between points of TR and the origin $0 \in \mathbf{R}^4$.

The approximation strategy followed based on a grid architecture along the lines of Chapter 6. The grid used is defined by the following divisions of each of the four axes of the state space:

$$
\begin{aligned}
[-2,-1.55,-1.1,-0.65,-0.2,0,0.2,0.65,1.1,1.55,2]: &\quad \text{grid of } x_1 \\
[-4,-3.1,-2.2,-1.3,-0.4,0,0.4,1.3,2.2,3.1,4]: &\quad \text{grid of } x_2 \\
[-90,-70,-50,-30,-10,10,30,50,70,90]\frac{\pi}{180}: &\quad \text{grid of } x_3 \\
[-90,-70,-50,-30,-10,10,30,50,70,90]\frac{\pi}{180}: &\quad \text{grid of } x_4
\end{aligned}
$$

The set of the vertices of the grid is the set of all possible quadruples $(\epsilon, \zeta, \eta, \theta)$ such that ϵ takes values in the top vector (grid of x_1) among the four vectors above, ζ in the second vector from the top (grid of x_2) and so on. Notice that the gridded area coincides with TR, by choice. The grid is a design tool, and it could have been chosen larger than the TR so as to account for trajectories which start off inside TR but go outside of it. It turns out that this is not necessary in the present example, and a stabilizing controller for the entire TR can be designed on the basis of the above grid.

8.1 Policy Evaluation, Cost Function Approximation, and Policy Update

We compute the directional derivative of the cost function J_{μ_0} as described in Section 6.1.2 at all vertices of the grid. Consider an *elementary subrectangle* Υ of the grid (defined in Section 6.1.3). This is a rectangle in $\mathbf{R}^4$, with 16 vertices and 8 facets. We denote the 16 vertices by $z_1, z_2, \ldots, z_{16}$, and the 8 facets by $S_1, S_2, \ldots, S_8$. We order the facets in such a way that S^i is facing opposite to S_{i+4}, for $i = 1, \ldots, 4$. To make the above description more precise, let $z_i = [z_i^1, z_i^2, z_i^3, z_i^4]^T$, for every $i = 1, \ldots, 16$, where z_i^1, z_i^2, z_i^3 and z_i^4 are the coordinates of z_i. Each facet is defined by a set of 8 vertices all of which share one common coordinate. For example, they may all have the same value of z_i^1. The opposite facet is defined by the rest of the vertices. Any two opposite facets do not share any common vertex. We define the pairs $\{S_1, S_5\}$, $\{S_2, S_6\}$, $\{S_3, S_7\}$, $\{S_4, S_8\}$ of opposite facets. As an example, let all vertices of S_1 have the same value of their first coordinate and let that value be ξ_a. If, for example, the vertex z_1 belongs to S_1, then $z_1 = [\xi_a, z_1^2, z_1^3, z_1^4]^T$. All vertices of S_5 then have the same value of their first coordinate, as well. Let that value be ξ_b, where $\xi_b \neq \xi_a$.

We then define the quantities $v_1, v_2, \ldots, v_8$ as follows:

$$v_i \triangleq \frac{1}{8} \sum_{\{j : z_j \in S_i\}} L_g J_{\mu_0}(z_j), \qquad i = 1, \cdots, 8. \tag{8.5}$$

The quantity v_i, for each i, is the mean value of $L_g J_{\mu_0}$ over all the vertices that belong to the facet S_i. The division over 8 in (8.5) is because each facet S_i contains 8 vertices. We return to the example of the facets S_1 and S_5, mentioned in the previous paragraph. Let $\xi_b > \xi_a$. Let $x = [x^1, x^2, x^3, x^4]^T$ belong to the elementary subrectangle Y, where x^1, x^2, x^3 and x^4 are the coordinates of x. We define the function $V_{1,5}(x)$ as

$$V_{1,5}(x) \triangleq \frac{1}{(\xi_b - \xi_a)} \left[x^1 (v_5 - v_1) - \xi_a v_5 + \xi_b v_1 \right], \qquad x \in \mathrm{Y}. \tag{8.6}$$

The function $V_{1,5}$ represents an averaging scheme between the two facets S_1 and S_5, and its definition (8.6) is equivalent to

$$\frac{V_{1,5}(x) - v_1}{x^1 - \xi_a} = \frac{v_5 - v_1}{\xi_b - \xi_a}, \qquad x \in \mathrm{Y}. \tag{8.7}$$

Similarly, we define the functions $V_{2,6}$, $V_{3,7}$ and $V_{4,8}$. Finally, the approximation of

$\tilde{L}_Y$ of $L_g J_{\mu_0}$ in the elementary subrectangle Y is defined as the mean value

$$\tilde{L}_Y(x) \triangleq \frac{1}{4}\left[V_{1,5}(x) + V_{2,6}(x) + V_{3,7}(x) + V_{4,8}(x)\right], \qquad x \in \mathrm{Y}. \tag{8.8}$$

This is a affine function of x of the form described in Chapter 6, for all x in Y.

Consider a value of the state x which does not belong to TR. Let Y be the elementary subrectangle whose Euclidean distance from x is the closest among all elementary subrectangles of the grid. We then approximate $L_g J_{\mu_0}(x)$ by $\tilde{L}_Y(x)$, which is defined by (8.8). This constitutes an extrapolation scheme outside TR.

Overall, the approximation of $L_g J_{\mu_0}$ is a piecewise affine function of the state, discontinuous at the facets of each subrectangle, denoted by $\tilde{L}J_{\mu_0}$.

8.2 Iteration 1

The controller $\mu_1(x)$ is defined as

$$\mu_1(x) = -\frac{1}{R}\tilde{L}J_{\mu_0}, \tag{8.9}$$

where $R = 0.1$ is the control input weight factor in the cost function. The closed loop system under μ_1 is not RAO-safe, but it is stable in the sense that its trajectories remain bounded, when the initial conditions belong to TR. In Figures (8.3) and (8.4) we show the trajectories of the position of the ball $x_1(t)$ and the beam angle $x_3(t)$, respectively, when the initial conditions are given by $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$ (the situation shown in Figure 8.2). The closed loop system under μ_1 moves towards the direction of decreasing beam angle, but during the transition between 90 and 0 degrees the ball rolls away from the origin under the influence of gravity. At the moment that the beam's angle becomes negative,

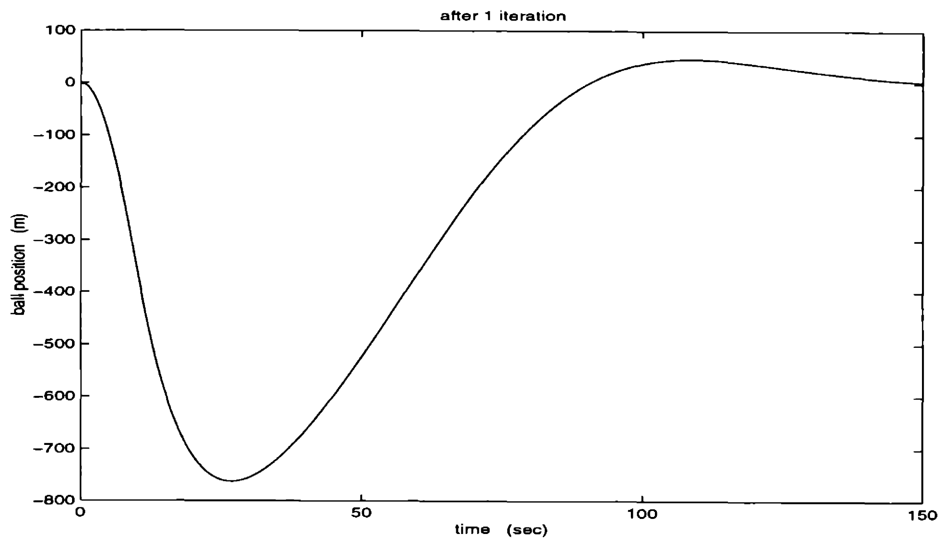

Figure 8.3: The trajectory of the ball's position $x_1(t)$ when the initial conditions are $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$, under policy μ_1.

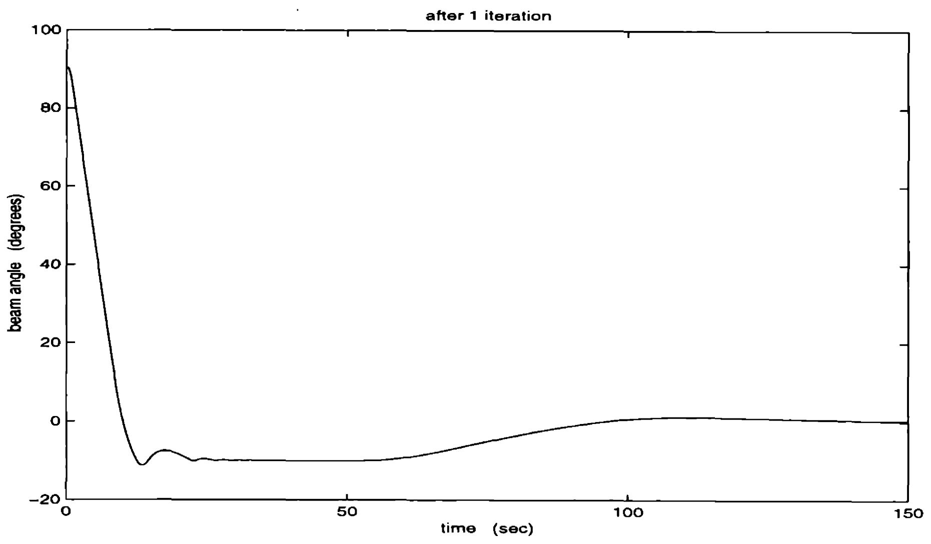

Figure 8.4: The trajectory of the beam's angle $x_3(t)$ when the initial conditions are $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$, under policy μ_1.

gravity tends to bring the ball towards the center of rotation. The ball decelerates and eventually starts moving towards the center of rotation. It overshoots, and the controller responds by forcing the beam angle x_3 to positive values. The closed loop system is asymptotically stable, but $x_1(t)$ experiences extremely large transients. The closed loop system clearly demonstrates the effectiveness of the approach suggested in Chapter 4 for stabilizing unstable systems.

Although the closed loop system under μ_1 clearly is not RAO-safe, we view it as a stable system with large transients and perform policy iteration so as to come up with a better controller, which results in better transients.

It turns out that if we use the same cost function and grid, and apply policy iteration to the closed loop system under μ_1, the next iterate is unstable. We thus look for a different cost function. We use a cost function (undiscounted) with state weight factor of the form $Q = \mathbf{diag}(Q_1, Q_2, Q_3, Q_4)$, with $Q_1 = 0.00004$, $Q_2 = 0.000002$, $Q_3 = 0.0004$, $Q_4 = 0.00008$ and control input weight factor $R = 10.0$. Notice that the matrix Q used in this iteration is the matrix Q used in the previous iteration divided by 10^5. The reason that this change is necessary to produce an improved controller has to do with the fact that the transients of the closed loop system under μ_1 are very large. A large value of Q translates into large values of the cost function. Consequently, any nonlinearity in the cost function is amplified and a finer grid is needed for approximation in order to account for the amplified nonlinearity. A smaller Q enables us to use the same grid as in the previous iteration with success. Note also that the input is weighted a lot more heavily than in the previous iteration. We choose to do so on the basis of the same argument. Since the approximate directional derivative of the cost function $\tilde{L}J_{\mu_1}$ is multiplied by $-1/R$ to give the new iterate μ_2, the factor $-1/R$ serves as a means of attenuating the approximation errors. At the same time, the torque input's level required to make the ball move towards the center of rotation, is not necessarily large. Indeed, it suffices to tilt the beam such that the

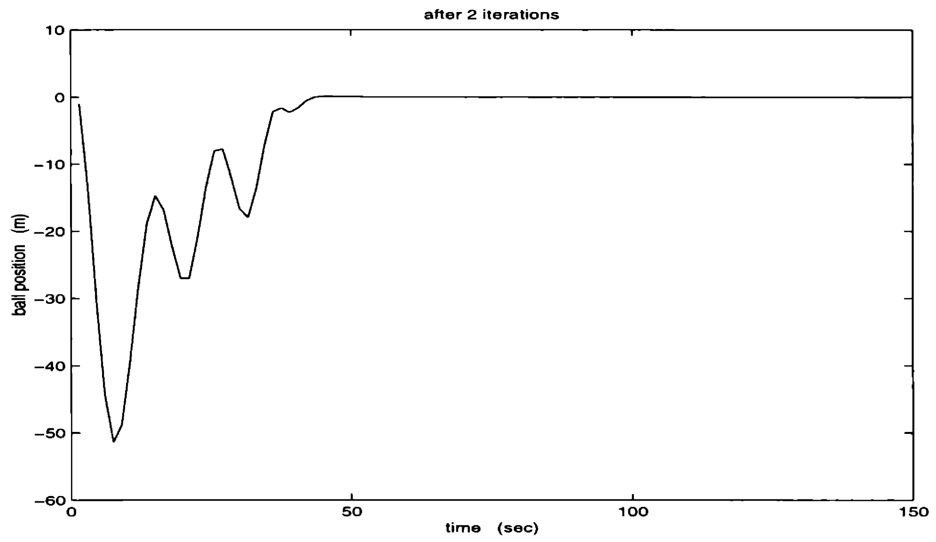

Figure 8.5: The trajectory of the ball's position $x_1(t)$ when the initial conditions are $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$, under policy μ_2.

ball is elevated with respect to the center of rotation. Then, gravity acting on the ball tends to bring it towards the center of rotation.

Finally, at the policy evaluation step of the closed loop system under μ_1, we simulate 150sec long trajectories with a sampling interval $\delta = 0.01$sec.

8.3 Iteration 2

The next iterate μ_2 results in a closed loop system that experiences improved transients compared to the system controlled by μ_1, for all initial conditions that belong to TR. In Figures (8.5) and (8.6) we show the trajectories of the position of the ball $x_1(t)$ and the beam angle $x_3(t)$, respectively, when the initial conditions are given by $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$ (the situation shown in Figure 8.2). It is clear that the transient of x_1 is quite improved in comparison to the transient of Figure 8.3. In fact, the controller used to produce the transients of Figures 8.5 and 8.6 is a hybrid of the controller generated via policy iteration and

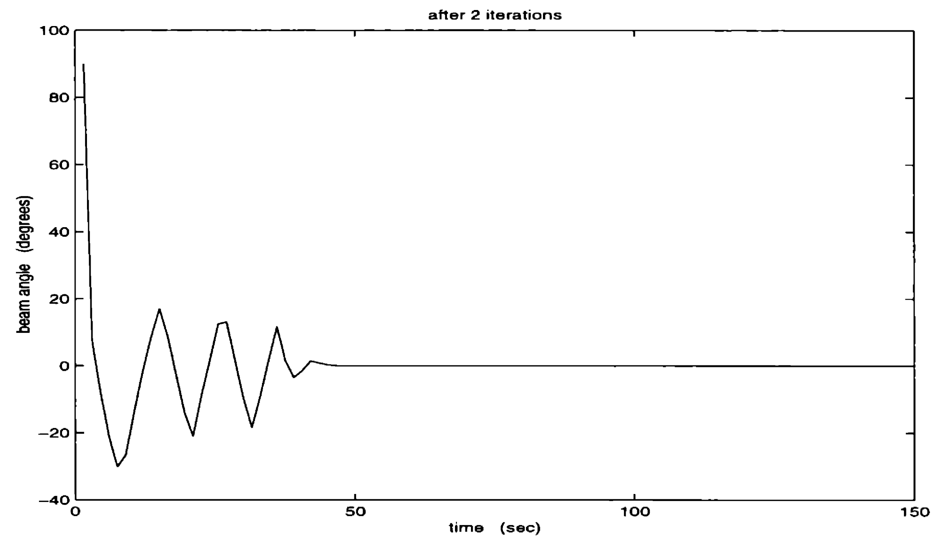

Figure 8.6: The trajectory of the beam's angle $x_3(t)$ when the initial conditions are $x_1(0) = -1$, $x_2(0) = 0$, $x_3(0) = 90\pi/180$ (90 degrees), $x_4(0) = 0$, under policy μ_2.

a linear controller which is designed on the basis of the linearized system around the equilibrium point. We switch to the linear controller when the state approaches the origin. This is because, if we use the nonlinear controller resulting from policy iteration, the trajectories do not converge asymptotically to 0, but they oscillate around it. We attribute this to the rather crude strategy that we use to interpolate between the vertices of the grid. A more sophisticated method has to be employed, as discussed in Chapter 6. However, the important thing is that the nonlinear controller forces the trajectories close enough to the origin such that the use of a linear controller can be used for fine tuning.

Finally, we note that the transients of the closed loop system under μ_2 compare quite well to the best available controllers for the beam and ball problem, in terms of the worst distance of the ball from the center of rotation during the transient, and in terms of the settling time. The only significant exception is that of the settling time reported in [49], which is around 20 seconds, in comparison to roughly 50 seconds of our controller. However, it is quite likely that better performance can be obtained by use of different parameters Q and R and/or a finer grid.

The conclusion from these two iterations is that approximate policy iteration is effective in stabilization problems, as well as performance improvement problems. No further iterations were performed, since the main objectives of this application have essentially already been achieved with first two iterations.

Chapter 9

Conclusions and Recommendations for Future Research

We proposed an approach for designing controllers for nonlinear, continuous time plants via approximate policy iteration. Existing methods for nonlinear control design typically work for special classes of systems and do not address the issue of performance. Approximate policy iteration is a (sub)optimal control design method which applies to general nonlinear systems, and it addresses performance by trying to minimize an appropriate cost functional. The design is semiglobal, that is, the closed loop system under a resulting controller is stable and meets the low cost criterion if the initial condition belongs to a bounded region which includes the equilibrium point. There is no restriction in selecting this bounded region as large as needed for any specific application.

In the case where an initial stabilizing controller (policy) is available, we approximate the cost function of the closed loop system under this initial controller. Approximation is necessary, since computing the exact cost function is an intractable task. Based on the approximate cost function, we come up with an updated controller. Improvement with respect to the cost function depends upon the quality of cost function approximation. Errors on the approximation of the cost function of the continuous time closed loop system are introduced in the following stages of the design process:

1. Discretization by means of first order Taylor expansion, with discretization interval δ (1.7).

2. Selection of state samples in which the actual cost function of the discretized system is computed.

3. Cost function approximation by means of an architecture which is tuned on the basis of the cost function samples.

Furthermore, our implementation of the policy update step is a departure from the update rule of the exact policy iteration algorithm. The update rule that we propose is easily implementable. However, it introduces an extra error which is shown to be small for a single iteration, if the discretization interval is small.

In Section 2.3, we developed bounds on the approximation error such that the updated controller after a single iteration is stable. We used Lyapunov function theory to derive these bounds. The allowed error bounds are large, as argued in Section 2.4. In Section 2.5, we developed bounds on the approximation error such that the continuous time closed loop system under the updated controller after a single iteration is improved with respect to the cost in comparison to the closed loop system under the controller before the iteration. Once again, the allowed error bounds are large, as argued in Section 2.6. The conclusion that we draw from Chapter 2 is that approximate policy iteration is feasible as a practical design tool, since limited approximation accuracy (within the error bounds) suffices to result in improved controllers.

In Chapter 3, we determined the worst case suboptimality of the closed loop systems under the resulting policies, as the iteration number increases. The worst case suboptimality bounds that we developed shrink as the cost function approximation error vanishes. However, suboptimality may not vanish even in the case of no approximation error, due to the closed form implementation of the policies which introduces a policy update step error, as discussed in Chapter 1.

In Chapter 4, we addressed the problem of designing a stabilizing controller for an unstable system. We proposed a modification of the system dynamics such that the modified system is an accurate representation of the original system in a selected region of the state space, the region of acceptable operation. We then posed an appropriate optimal control problem for the modified system and showed that the optimal controller is a stabilizing controller for the original system. The system dynamics modification enables the use of approximate policy iteration for determining a stabilizing controller. From a real world application point of view, the modification is justified by the fact that in practice we are typically interested in stable operation in a bounded region around the equilibrium point. This region is specified by safety limits or precision tolerances. The development in Chapter 4 thus shows feasibility of the approximate policy iteration design approach for designing stabilizing controllers for unstable systems.

The confidence results of Chapters 2-4 show that the bounds on the allowed approximation errors can be large. Next, we look for ways to simplify the task of cost function approximation. This is a potentially nontrivial task despite the large allowed errors, especially in large dimensions. In Chapters 5-6, we proposed two natural approaches for cost function approximation, with the potential of simplifying the task.

We suggested that insight into the structure of the dynamics can be used in order to partially determine the form of the actual cost function. In Chapter 5, we proposed a simplification of the approximation task in the specific class of systems whose dynamics are linear with respect to some of the states. In conjuction with a quadratic cost function, a natural cost function approximation is quadratic in this part of the state.

For general systems, we suggested a grid-based approximation strategy, which consists of computing the appropriate directional derivatives of the cost function at the vertices of a state space grid and approximating via interpolation throughout the remainder of the state space. The feasibility results of Chapter 2 are in favor of a grid approach, since they suggests that not too fine a grid could suffice in many cases. Furthermore, in Chapter 6 it was shown that the error bounds for stability of the next iterate developed in Chapter 2 can be checked with the help of a convex feasibility test, which is enabled by a grid-based, piecewise affine approximation of the directional derivative.

In conclusion, Chapters 5 and 6 include our ideas for enhancing practicality of approximate policy iteration as a nonlinear control design approach.

Finally, the method was evaluated via two design examples. In Chapter 7, we successfully applied the method to a stabilized missile control problem. The control objective is step command following of the normal acceleration output. The state was augmented by an integrator state and a reference command state to ensure perfect following for all sizes of the step. The augmented state was 4-dimensional. The selected approximation architecture was quadratic in 3 of the states, along the lines of the simplification suggested in Chapter 5. The result was cost reduction by roughly a factor of 8 in 22 iterations. In Chapter 8, we applied the method to the problem of stabilization of a beam-and-ball system. The system dynamics were modified along the lines of Chapter 4, and a grid architecture was used, along the lines of Chapter 6. The system was stabilized in a single iteration, whereas a second policy iteration resulted in a controller of improved performance.

9.1 Recommendations for future research

We suggest several directions which can be further pursued on this nonlinear control design approach. Some of the suggested topics aim at enhancing practicality of the method by making the cost function approximation task easier:

1. The directional derivative of the actual cost function is given by the solution of a system of linear parameter-varying differential equations, as shown in Chapter 2. As discussed in Chapter 6, this fact could potentially be used to formulate a convex optimization problem whose solution would be an upper bound on the approximation error of a grid-based, piecewise linear approximation architecture presented in Chapter 6. In conjuction with the test developed in Chapter 6 for determining an upper bound on the *allowed* approximation error, such a test would provide a systematic method for checking the sufficient conditions for stability of the next iterate. According to the result of these tests, it could be determined whether a grid is sufficiently fine. Along the same lines, such tests could be developed for checking the sufficient conditions for cost improvement after a single iteration.

2. In Chapter 8 we described a linear interpolation scheme between the values of the cost function directional derivatives at the vertices of a grid. As argued in Chapter 6, it is not possible to find a linear interpolation which matches the values at the vertices, or such that the values in the interior of the elementary subrectangle range between the minimum and the maximum of the vertex values. An alternative interpolation strategy with the above properties could be implemented if the state space is divided into simplices instead of rectangles. A different suggestion is to use polynomial interpolation of higher even degrees between the vertices of the rectangle, instead of linear interpolations. In the latter

case, developing new convex tests for checking the soundness of approximation would constitute a future research task.

Another suggested direction is that of *robust nonlinear control.* Each policy iteration results into a Lyapunov function for the system (the cost function associated with the closed loop under the corresponding policy). Each of these functions is not available in closed form, but it is approximated. Then, the robustness properties of the closed loop system can be evaluated by means of the approximated Lyapunov function, along the lines of [25]. Policy iteration can be repeated until satisfactory robustness is achieved.

Bibliography

[1] D. Aeyels. Local and global controllability for nonlinear systems. *Syst. Control Letters*, 5:19–26, October 1984.

[2] V.M. Alekseev, V.M. Tikhomirov, and S.V. Fomin. *Optimal Control.* Contemporary Soviet Mathematics. Consultants Bureau, New York, 1987.

[3] P. Apkarian and P. Gahinet. A convex characterization of gain-scheduled $\mathbf{H}_\infty$ controllers. *IEEE Trans. Aut. Control*, 40(5):853–864, May 1995.

[4] P. Apkarian, P. Gahinet, and G. Becker. Self-scheduled $\mathbf{H}_\infty$ control of linear parameter-varying systems: a design example. *Automatica*, 31(9):1251–1261, September 1995.

[5] Z. Artstein. Stabilization with relaxed controls. *Nonlinear Analysis, Theory, Methods & Applications*, 7(11):1163–1173, 1983.

[6] G. Becker and A. Packard. Robust performance of linear parametrically varying systems using parametrically-dependent linear feedback. *Syst. Control Letters*, 23(3):205–215, September 1994.

[7] R. Bellman. *Stability Theory of Differential Equations.* McGraw-Hill, 1953.

[8] R. Bellman. *Dynamic Programming.* Princeton Univ. Press, Princeton, NJ, 1957.

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

[10] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, Belmont, MA, 1995.

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

[12] J. H. Blakelock. *Automatic Control of Aircraft and Missiles.* John Wiley & Sons, second edition, 1991.

[13] A. M. Bloch and N. H. McClamroch. Control of mechanical systems with classical nonholonomic constraints. In *Proc. IEEE Conf. on Decision and Control*, pages 201–205, Tampa, Florida, December 1989.

[14] A. M. Bloch, M. Reyhanoglu, and N. H. McClamroch. Control and stabilization of nonholonomic dynamic systems. *IEEE Trans. Aut. Control*, 37(11):1746–1757, November 1992.

[15] C. Boussios and E. Feron. Estimating the conservatism of Popov's criterion for real parametric uncertainties. *Syst. Control Letters*, 31:173–183, 1997.

[16] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *SIAM Studies in Applied Mathematics.* SIAM, 1994.

[17] S. J. Bradtke, B. E. Ydstie, and A. G. Barto. Adaptive linear control using policy iteration. In *Proc. American Control Conf.*, pages 3475–3479, Baltimore, MD, 1994.

[18] R. W. Brockett. Asymptotic stability and feedback stabilization. In R. W. Brockett, R. S. Millman, and H. J. Sussman, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhauser, Boston, MA, 1983.

[19] T. Burg, D. Dawson, and P. Vedagarbha. A redesigned DCAL controller without velocity measurements: theory and demonstration. *Robotica*, 15(3):337–346, May 1997.

[20] C.-P. Chao and P. M. Fitzsimons. Stabilization of a large class of nonlinear systems using conic sector bounds. *Automatica*, 33(5):945–953, May 1997.

[21] M. G. Crandall and P. L. Lions. Two approximations of solutions of Hamilton-Jacobi equations. *Mathematics of Computation*, 43:1–19, 1984.

[22] K. B. Datta. *Matrix and Linear Algebra.* Prentice-Hall, New Delhi, India, 1991.

[23] 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.

[24] T.M. Flett. *Differential Analysis.* Cambridge University Press, Cambridge, UK, 1980.

[25] R. A. Freeman and P. V. Kokotovic. *Robust Nonlinear Control Design.* Systems and Control: Foundations and Applications. Birkhauser, Boston, 1996.

[26] S. Torkel Glad. On the gain margin of nonlinear and optimal regulators. *IEEE Trans. Aut. Control*, 29(7):615–620, July 1984.

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

[28] T. H. Gronwall. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Math.*, 20(2):292–296, 1919.

[29] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer-Verlag, New York, NY, 1985.

[30] P. Hartman. *Ordinary Differential Equations.* John Wiley & Sons, New York, 1964.

[31] J. K. Hedrick and S. Gopalswamy. Nonlinear flight control design via sliding methods. *AIAA Journal of Guidance, Navigation and Control*, 13(5), 1990.

[32] R. Hermann and A. J. Krener. Nonlinear controllability and observability. *IEEE Trans. Aut. Control*, 22(5):728–740, October 1977.

[33] H. Hermes. Control systems which generate decomposable Lie algebras. *Journal of Differential Equations*, 44(2):166–187, 1982.

[34] O. Hernandez-Lerma and M. Munoz de Ozak. Discrete-time MCPs with discounted unbounded costs: Optimality criteria. *Kybernetika (Prague)*, 28:191–212, 1992.

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

[36] R. A. Howard. *Dynamic Programming and Markov Processes.* MIT Technology Press, Cambridge, MA, 1960.

[37] L. R. Hunt, R. Su, and G. Meyer. Global transformations of nonlinear systems. *IEEE Trans. Aut. Control*, 28:24–31, 1983.

[38] A. Isidori. A remark on the problem of semiglobal nonlinear output regulation. *IEEE Trans. Aut. Control*, 42(12):1734–1738, December 1997.

[39] J. Kaloust, C. Ham, and Z. Qu. Nonlinear autopilot control design for a 2-DOF helicopter model. In *IEE Proceedings-Control Theory and Applications*, volume 144, pages 612–616, November 1997.

[40] H. K. Khalil. Robust servomechanism output feedback controllers for feedback linearizable systems. *Automatica*, 30(10):1587–1599, October 1994.

[41] H. K. Khalil. *Nonlinear Systems.* Prentice Hall, Upper Saddle River, second edition, 1996.

[42] H. J. Kushner and P. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time.* Springer-Verlag, New York, NY, 1992.

[43] S. Lefschetz. *Differential Equations: Geometric Theory.* Interscience Publishers, New York, second edition, 1963.

[44] W. Lin. Feedback stabilization of general nonlinear control systems: A passive system approach. *Syst. Control Letters*, 25:41–52, 1995.

[45] C. R. F. Maunder. *Algebraic Topology.* The New University Mathematics Series. Van Nostrand Reinhold Company, London, 1970.

[46] M. W. McConley. *A Computationally Efficient Lyapunov-Based Procedure for Control of Nonlinear Systems with Stability and Performance Guarantees.* PhD thesis, M.I.T., Cambridge, MA, 1997.

[47] M. W. McConley, B. D. Appleby, M. A. Dahleh, and E. Feron. A control Lyapunov function approach to robust stabilization of nonlinear systems. In *Proc.*

34th Annual Allerton Conf. on Communication, Control and Computing, pages 372–381, Allerton House, Monticello, Illinois, October 1996.

[48] Y. Nesterov and A. Nemirovsky. *Interior-point polynomial methods in convex programming*, volume 13 of *SIAM Studies in Applied Mathematics*. SIAM, 1994.

[49] R. Olfati-Saber and A. Megretski. Controller design for the beam-and-ball system. *Submitted to the 1998 IEEE Conference on Decision and Control*, 1998.

[50] A. Packard. Gain scheduling via linear fractional transformations. *Syst. Control Letters*, 22(2):79–92, February 1994.

[51] M. H. Protter and Jr. C. B. Morrey. *A First Course in Real Analysis*. Springer-Verlag, New York, 1977.

[52] R. Reichert. Modern robust control for missile autopilot design. In *Proceedings of the American Control Conference*, pages 2368–2373, San Diego, CA, June 1990.

[53] W. J. Rugh. Analytical framework for gain scheduling. *IEEE Control Syst. Mag.*, 11(1):79–84, January 1991.

[54] A. Saberi, P. V. Kokotovic, and H. J. Sussmann. Global stabilization of partially linear composite systems. *SIAM J. on Control and Optimization*, 28:1491–1503, 1990.

[55] R. Sepulchre, M. Jankovic, and P. Kokotovic. *Constructive Nonlinear Control.* Springer-Verlag, London, 1997.

[56] S. M. Shahruz and S. Behtash. Design of controllers for linear parameter-varying systems by the gain scheduling technique. *Journal of Mathematical Analysis and Applications*, 168(1):195–217, July 1992.

[57] J. S. Shamma and M. Athans. Analysis of gain scheduled control for nonlinear plants. *IEEE Trans. Aut. Control*, 35(8):898–907, August 1990.

[58] J. S. Shamma and M. Athans. Guaranteed properties of gain scheduled control for linear parameter-varying plants. *Automatica*, 27(3):559–564, May 1991.

[59] J. S. Shamma and J. R. Cloutier. Gain-scheduled missile autopilot design using linear parameter varying transformations. *AIAA Journal of Guidance, Control and Dynamics*, 16(2):256–263, 1993.

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

[61] J. J. Slotine. Sliding controller design for nonlinear systems. *International Journal of Control*, 40(2):421–434, 1984.

[62] J. J. Slotine and W. Li. *Applied Nonlinear Control.* Prentice-Hall, Englewood Cliffs, NJ, 1991.

[63] E. D. Sontag. Feedback stabilization of nonlinear systems. In M. A. Kaashoek, J. H. van Schuppen, and A. C. M. Ran, editors, *Robust Control of Linear Systems and Nonlinear Control*, pages 61–81. Birkhauser, Cambridge, MA, 1990.

[64] C. Y. Su and Y. Stepanenko. Sliding mode control of nonholonomic mechanical systems: underactuated manipulators case. In A. J. Krener and D. Q. Mayne, editors, *Nonlinear Control Systems Design 1995*, pages 565–569. Pergamon, 1995. Published for the IFAC.

[65] R. Su. On the linear equivalents of nonlinear systems. *Syst. Control Letters*, 2:48–52, 1982.

[66] H. J. Sussman. Lie brackets and local controllability: a sufficient condition for scalar-input systems. *SIAM J. on Control and Optimization*, 21(5):686–713, 1983.

[67] H. J. Sussmann. Semigroup representations, bilinear approximation of input-output maps, and generalized inputs. In G. Marchesini and S. K. Mitter, editors, *Proceedings of the International Symposium on Mathematical Systems Theory*, pages 172–191, Udine, Italy, June 1976.

[68] R. S. Sutton and A. G. Barto. *Reinforcement Learning.* MIT Press, Cambridge, MA, 1998.

[69] A. Teel and L. Praly. Tools for semiglobal stabilization by partial state and output feedback. *SIAM J. on Control and Optimization*, 33(5):1443–1488, September 1995.

[70] A. R. Teel. Semi-global stabilization of the 'ball and beam' using 'output' feedback. In *Proceedings of 1993 American Control Conference*, volume 3, pages 2577–2581, San Francisco, CA, 1993.

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

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

[73] J. N. Tsitsiklis and M. Athans. Guaranteed robustness properties of multivariable nonlinear stochastic optimal regulators. *IEEE Trans. Aut. Control*, 29(8):690–696, August 1984.

[74] V. I. Utkin. *Sliding Modes and their Applications.* Mir, Moscow, 1978.

[75] P. P. Varaiya and R. Liu. Bounded-input bounded-output stability of nonlinear time-varying differential systems. *SIAM J. on Control*, pages 698–704, 1966.

[76] M. Vidyasagar. *Nonlinear Systems Analysis.* Prentice-Hall, Englewood Cliffs, second edition, 1993.

[77] B. Vik and T. I. Fossen. Semiglobal exponential output feedback control of ships. *IEEE Transactions on Control Systems Technology*, 5(3):360–370, May 1997.

[78] J. Wang and W. J. Rugh. Parameterized linear systems and linearization families for nonlinear systems. *IEEE Trans. Circuits Syst.*, 34(6):650–657, June 1987.

[79] 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.

[80] D. A. White and D. A. Sofge, editors. *Handbook of Intelligent Control.* Van Nostrand Reinhold, New York, NY, 1992.

[81] V. A. Yakubovich. The $\mathcal{S}$-procedure in non-linear control theory. *Vestnik Leningrad Univ. Math.*, 4:73–93, 1977.