Approximate Dynamic Programming with Applications

Wernrud, Andreas

2008

Document Version:
Publisher's PDF, also known as Version of record

Approximate Dynamic Programming with Applications

Approximate Dynamic Programming with Applications

Andreas Wernrud

Department of Automatic Control
Lund University
Lund, 2008

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

ISSN 0280–5316
ISRN LUTFD2/TFRT--1082--SE

Printed in Sweden by Media-Tryck.
Lund 2008

Till Erika

Abstract

This thesis studies approximate optimal control of nonlinear systems. Particular attention is given to global solutions and to the computation of approximately optimal feedback controllers. The solution to an optimal control problem is characterized by the optimal value function. For a large class of problems the optimal value function must satisfy a Hamilton-Jacobi-Bellman type equation. Two common methods for solving such equations are policy iteration and value iteration. Both these methods are studied in this thesis.

An approximate policy iteration algorithm is presented for both the continuous and discrete time settings. It is shown that the sequence produced by this algorithm converges monotonically towards the optimal value function.

A multivariate polynomial relaxation algorithm is proposed for linearly constrained discrete time optimal control problems with convex cost.

Relaxed value iteration is studied for constrained linear systems with convex piecewise linear cost. It is shown how an explicit piecewise linear control law can be computed and how the resulting look-up table can be reduced efficiently.

The on-line implementation of receding horizon controllers, even for linear systems, is usually restricted to systems with slow dynamics. One reason for this is that the delay between measurement and actuation introduced by computing the control signal on-line can severely degrade systems with fast dynamics. A method to improve robustness against such delays and other uncertainties is presented.

A case study on the control of DC–DC converters is given. Feasibility of a Relaxed Dynamic Programming algorithm is verified by synthesizing controllers for both a step-down converter and a step-up converter. The control performance is evaluated both in simulations and in real experiments.

Acknowledgments

I would like to use this small space to give my most sincere thanks to the people who have helped me complete this thesis.

As my thesis supervisor, Anders Rantzer has been excellent. He has provided guidance at key moments during my work while also allowing me to work independently most of the time, for which I am especially grateful. I am also grateful for the suggestions on research topics he gave me a couple of years ago. Some of those topics are covered later in this thesis.

Thanks to Anders Robertsson for answering all kinds of questions in my early days at the department. I appreciate the feedback provided by Peter Alriksson on some of my work, it was very helpful. Thanks to Karl Mårtensson for our collaboration on DMPC, hopefully we will continue with that work.

A special thanks to Stéphane Velut for many valuable discussions. Also, thanks for sharing your climbing skills with me.

Several persons, Anders Rantzer, Anders Robertsson, Peter Alriksson and Brad Schofield, have read parts of drafts of this thesis. They have tracked down typos and given me suggestions on alternative formulations. Although I did ignore some of those suggestions, the ones I used have definitely improved the presentation of my work. Thanks a lot!

The Department of Automatic Control is a great place to do graduate studies. Not only do we have skilled researchers, we also have skilled secretaries, Eva, Britt-Marie and Agneta and also a competent technical staff, Leif, Rolf and Anders. Thank you all for making daily work easier!

I thank my collaborators on the benchmark problems in the HYCON project. In particular, I would like to thank Ulf Jönsson, Stefan Almér, Andreas Beccuti and Sébastien Mariethoz for editing our papers. A special thanks to the ETH group for providing me and the other groups with the opportunity to do experiments at ETH.

I gratefully acknowledge the financial support for this work, provided

by the Swedish Research Council and HYCON.

I thank my family and friends for all the support they have given me and still do today. You all mean very much to me.

Finally, the person whom I want to thank the most is Erika. I am probably too lucky to have you by my side.

Andreas

Contents

Preface

Motivation

Optimal control methods can be a very helpful tool when solving synthesis problems. It allows the control designer to pose the synthesis problem as a mathematical optimization problem. The solution to the resulting optimization problem can sometimes be characterized and computed using systematic methods.

There are two distinct types of optimal control problems, open loop and closed loop problems. The solution to an open loop problem is an optimal control trajectory, starting from a fixed initial condition. The solution to a closed loop problem is a feedback controller, i.e. a function mapping the state to the optimal control value. A solution to an open loop problem can usually be obtained by solving a two point boundary ordinary difference or differential equation, derived by applying Pontryagins Minimum Principle, see [Leitmann, 1981]. It is much more complicated to solve a closed loop problem since the information represented by the optimal value function is essentially equal to the information obtained by solving a two point boundary ordinary difference or differential equation from each point in state space. It turns out that the optimal value function must satisfy a Hamilton-Jacobi-Bellman(HJB) type equation, a nonlinear partial difference or differential equation, see [Bellman, 1957; Leitmann, 1981; Fleming and Soner, 1993]. Closed loop problems are also referred to as global methods since each point in the state space is considered.

Even though modern optimal control has been an active research topic since the 1950s, there has been little progress in finding constructive solution techniques for this equation. Almost all proposed techniques are based on gridding continuous variables. A grid-based solution technique means that candidate solutions are parametrized by their function values

on a finite number of points. A major advantage with this parametrization is that it can be applied to almost any type of problem. However, since the grid size scales exponentially with dimension, gridding is usually not practical for problems with more than a few states. For problems with few states it can be a good choice, see [Grüne, 1997]. On the other hand, methods that use a less flexible parametrization, e.g. a polynomial, can only be applied to problems with some special structure. Moreover, using a less flexible parametrization also means that exact solutions are less likely to be found. It is natural to formulate an approximate solution strategy. In this thesis, the exact problem is approximated be replacing equality constraints by inequalities in such a way that the approximation has certain attractive properties. Moreover, parametrization are chosen so that all required inequalities can be verified using either semidefinite programming or linear programming.

Another approach to optimal control is Model Predictive Control or Receding Horizon Control (RHC). The essential difference between RHC and the more direct approach discussed above is in terms of implementation. In RHC all necessary computations are performed on-line, thus avoiding the difficulties involved in the computation of the optimal value function. On the other hand, the on-line computation required in RHC can sometimes be prohibitive e.g. for systems with fast dynamics. A method to approach this problem is presented in this thesis.

Outline and Contributions

This section contains an outline of the thesis and a summary of the contributions.

Chapter 1: Background

This first chapter gives a short introduction to the vast subject of optimal control. Emphasis is placed on parts of this theory that are most relevant for this thesis. Also, techniques for non-negativity verification of polynomials are discussed.

Chapter 2: Approximate Policy Iteration

An approximate policy iteration algorithm for both the continuous and discrete time setting is given in this section. It is shown that the sequence produced by this algorithm converges monotonically to the optimal value function.

The theory is applied to optimal control of polynomial input affine systems with quadratic penalty on control values.

Related Publications

Wernrud, A. (2007): "Strategies for computing switching feedback controllers." In *American Control Conference*. New York City, USA.

Wernrud, A. and A. Rantzer (2005): "On approximate policy iteration for continuous-time systems." In *The 44th IEEE Conference on Decision and Control and European Control Conference ECC.* Seville, Spain.

Chapter 3: Value Iteration With Polynomial Parametrization

In this chapter we consider two types of optimal control problems. We propose solutions based on relaxed value iteration with multivariate polynomials as value function parametrization. First we consider constrained control of discrete time systems with continuous control space. To be able to perform value iteration we impose certain convexity assumptions and propose a semi-grid-based technique to verify a required inequality.

The second type of problem we consider is control of discrete time systems with discrete control space, i.e. a switching system. We propose a simple state weighting relaxation technique.

Related Publications

Wernrud, A. (2006): *Computation of approximate value functions for constrained control problems*. Proc. of the 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto, Japan.

Wernrud, A. (2007): "Strategies for computing switching feedback controllers." In *American Control Conference*. New York City, USA.

Chapter 4: Constrained Control of Linear Systems

Relaxed value iteration is studied for constrained linear systems with piecewise linear step cost. It is shown how an explicit piecewise linear control law can be computed and how the resulting look-up table can be reduced efficiently.

Related Publications

Wernrud, A. (2008): *On constrained optimal control of linear systems with piecewise linear step cost*. Submitted.

Chapter 5: Dynamic Model Predictive Control

The on-line implementation of receding horizon controllers, even for linear systems, is usually restricted to systems with slow dynamics. One reason for this is that the delay between measurement and actuation introduced by computing the control signal on-line can degrade systems with fast dynamics. Preliminary results and ideas on how to improve robustness against such delays and other uncertainties are presented.

Related Publications

Mårtensson, K., A. Wernrud (2008): "Dynamic Model Predictive Control" To appear at the 17th IFAC World Congress.

Chapter 6: Control of DC-DC converters: A Case Study

Power electronic circuits such as the switched mode dc-dc converters represents a good entry point for the investigation of the control design and performance benefits that can be brought by hybrid and optimal control techniques. A case study on the control of DC–DC converters is given. Feasibility of a Relaxed Dynamic Programming algorithm is verified by synthesizing controllers for both a step-down converter and a step-up converter.

Related Publications

Almer, S., H. Fujioka, U. Jönsson, C. Y. Kao, D. Patino, P. Riedinger, T. Geyer, A. Beccuti, G. Papafotiou, M. Morari, A. Wernrud, A. Rantzer (2007): *Hybrid Control Techniques for Switched-Mode DC–DC Converters, Part I: The Step-Down Topology*. Proc. of American Control Conference.

Beccuti, A., G. Papafotiou, M. Morari, S. Almer, H. Fujioka, U. Jönsson, C. Y. Kao, A. Wernrud, A. Rantzer, M. Baja, H. Cormerais, J. Buisson (2007): *Hybrid Control Techniques for Switched-Mode DC–DC Converters, Part II: The Step-Up Topology*. Proc. of American Control Conference.

Mariéthoz, S., S. Almér, A. Beccuti, D. Patino, A. Wernrud, T. Geyer, H. Fujioka, U. Jönsson, C.-Y. Kao, M. Morari, G. Papafotiou, A. Rantzer, and P. Riedinger (2008a): "Evaluation of four hybrid control techniques for the synchronous step down buck DC–DC converter." To be submitted.

Mariéthoz, S., S. Almér, B. Mihai, A. G. Beccuti, A. Wernrud, H. Fujioka, U. Jönsson, C.-Y. Kao, H. Cormerais, J. Buisson, G. Papafotiou, M. Morari, and A. Rantzer (2008b): "Comparative assessment of hybrid control techniques for the boost DC–DC converter." To be submitted.

1

Background

1.1 Introduction

This chapter is meant to be a background and brief summary to the problems studied later in this thesis. Detailed treatments of optimal control can be found in [Bellman, 1957; Leitmann, 1981; Cesari, 1983; Fleming and Soner, 1993; Bertsekas, 2000; Bardi and Capuzzo-Dolcetta, 1997].

The optimal control problem is stated in both continuous and discrete time. The classical theorem showing necessity of the Hamilton-Jacobi-Bellman equation is stated and in fact proved. A proof is given because it is instructive. It shows the essence of the Dynamic Programming approach to solve optimal control problems, i.e. the principle of optimality turns a time trajectory optimization problem into a pointwise in state space optimization problem.

The key ideas and results of Relaxed Dynamic Programming are reviewed, these ideas are used in Chapters 3–4.

Techniques for verification that a polynomial is non-negative are introduced and some illustrating examples are given.

1.2 Optimal Control

The essential question answered by the theory of optimal control is how to best choose the control input to a dynamical system, where "best" is defined to be the input that results in minimal state/input trajectory cost.

Our interest is in global solutions, i.e. we would like to find the best input for each initial condition from a prescribed set of states, not just a trajectory from a fixed initial condition.

Global solutions to optimal control problems can be characterized, and sometimes computed, by using Dynamic Programming. For discrete time problems the characterization is given in terms of the solution to the Bellman equation, a nonlinear difference equation. The corresponding equation in continuous time, the Hamilton-Jacobi-Bellman equation, is a nonlinear partial differential equation. Both these equations are direct consequences of the so called Principle of Optimality:

From any point on an optimal trajectory, the remaining trajectory is optimal for the corresponding problem initiated at that point.

The HJB-Equation In Continuous Time

Consider a continuous time system

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

with $(x,u) \in \mathbb{X} \times \mathbb{U} \subset \mathbb{R}^n \times \mathbb{R}^m$. It is assumed that $0 = f(0,0)$. We denote the instantaneous cost function by $l(x,u)$. The instantaneous cost function is assumed to be positive definite, in other words $l(x,u) > 0$ if $(x,u) \neq 0$ and $l(0,0) = 0$. The total cost is defined as

$$V(x_0) = \int_0^\infty l(x,u)dt \tag{1.2}$$

where the initial state $x_0 := x(0) \in \mathbb{X}$ is given. The optimal control problem we consider is defined by

$$V^*(x_0) = \inf_{u(\cdot)} V \quad \text{such that (1.1) is satisfied.} \tag{1.3}$$

where $u(\cdot)$ denote a time function.

The Principle of Optimality can be encoded in the Dynamic Programming Equation

$$V^*(x_0) = \inf_{u(\cdot)}\{\int_0^\epsilon l(x(s),u(s))ds + V^*(x(\epsilon))\}$$

Throughout this thesis, the gradient of a differentiable function f will be denoted by $Df = \frac{\partial f}{\partial x}$. The gradient is defined to be a column vector.

Using this equation, we can prove the following classical result which shows that the optimal value function must satisfy the HJB-equation, at least if it is differentiable.

THEOREM 1.1
Suppose that u^* attains the minimum in (1.3) and that the corresponding optimal cost V^* is differentiable, then V^* satisfies the HJB-equation

$$\min_u \{DV^* \cdot f(x,u) + l(x,u)\} = 0 \tag{1.4}$$

and

$$u^*(t) = \mu(x^*(t)) = \text{argmin}_u \{DV^* \cdot f(x,u) + l(x,u)\} \tag{1.5}$$

□

PROOF 1.1
Let $\epsilon > 0$ and $x_0 = x(0)$ be any initial condition

$$\begin{aligned} V^*(x_0) &= \min_{u(\cdot)} \{ \int_0^\epsilon l(x(s),u(s))ds + V^*(x(\epsilon))\} \\ &= \min_{u(\cdot)} \{ \int_0^\epsilon l(x(s),u(s))ds + V^*(x(0) + f(x(0),u(0))\epsilon + o(\epsilon))\} \\ &= \min_{u(\cdot)} \{ \int_0^\epsilon l(x(s),u(s))ds + V^*(x(0)) + DV^* \cdot f(x(0),u(0))\epsilon + o(\epsilon)\} \end{aligned}$$

dividing by ϵ and then sending $\epsilon \to 0$ we get

$$\min_{u(0)} \{DV^* \cdot f(x(0),u(0)) + l(x(0),u(0))\} = 0$$

The result follows since $x(0) \in \mathbb{X}$ was arbitrary. □

The second equality follows from writing $x(\epsilon) = x(0) + f(x(0),u(0))\epsilon + o(\epsilon)$, the last step uses the assumption that V^* is differentiable and therefore has a first order Taylor expansion at every state.

The next result, which is also a classical one, shows that the HJB-equation provides a sufficient condition for optimality. The result is sometimes called the Verification Theorem of Dynamic Programming.

THEOREM 1.2
Suppose V is positive definite and satisfies the HJB-equation

$$\min_u \{DV \cdot f(x,u) + l(x,u)\} = 0 \tag{1.6}$$

Define

$$\mu(x(t)) = \text{argmin}_u \{DV \cdot f(x,u) + l(x,u)\} \tag{1.7}$$

If the solution to $\dot{x} = f(x,\mu(x))$ does not leave the domain of V^* then

$$V = V^*, \quad u^*(t) = \mu(x(t)) \tag{1.8}$$

□

PROOF 1.2
Integrating both sides of the equation $-DV \cdot f(x, \mu(x)) = l(x, \mu(x))$ gives

$$V(x(0)) - V(x(t)) = \int_0^t l(x, \mu(x))dt \leq V(x(0)) \tag{1.9}$$

If $\hat{u}$ is any stabilizing, other than $\mu(x)$, control function then

$$\int_0^t l(x, \mu(x))dt \leq V(x(0)) - V(x(t)) \leq \int_0^t l(x, \hat{u})dt \tag{1.10}$$

in the limit this becomes

$$\int_0^\infty l(x, \mu(x))dt \leq V(x(0)) \leq \int_0^\infty l(x, \hat{u})dt \tag{1.11}$$

which is an equality if $\hat{u} = \mu(x)$. □

We give a simple example which we will use in Chapter 2 to illustrate an approximation algorithm.

EXAMPLE 1.1
Consider the cost function

$$V(x_0) = \int_0^\infty x^2 + x^4 + u^2 dt$$

We would like to minimize this cost under the linear dynamic constraint

$$\dot{x} = -x + u$$

In this case the HJB-equation reduces to

$$\begin{aligned} 0 &= \min_u \{DV \cdot f(x,u) + l(x,u)\} \\ &= \min_u \{DV \cdot (-x+u) + x^2 + x^4 + u^2\} \\ &= 2x^2 + x^4 - \frac{1}{4}(DV + 2x)^2 \end{aligned}$$

thus

$$DV = -2x \pm 2|x|\sqrt{2 + x^2}$$

integrating this equation and using that V^* must be positive definite we find

$$V(x) = -x^2 + 2\frac{(2+x^2)^{3/2} - 2\sqrt{2}}{3}$$

and

$$\mu(x) = x - x\sqrt{2+x^2}$$

Theorem 1.2 shows that $V = V^*$ and $\mu = \mu^*$. □

Such examples, when equation (1.6) can be solved explicitly, are rare in higher dimensions. Moreover, both these theorems are useless for many problems since the optimal value function need not be differentiable. The Dynamic Programming approach to optimal control is still useful even if the optimal value function is not differentiable. The most well known extension to the classical theory is when the value function is allowed to be a so called viscosity solution to a HJB-type equation, see [Fleming and Soner, 1993; Bardi and Capuzzo-Dolcetta, 1997]. However, it is still equally difficult to solve the HJB-equation in this more general theory.

The HJB-Equation In Discrete Time

The notation in the discrete time setting is quite similar to the continuous case. The dynamic constraint now takes the form

$$x(k+1) = f(x(k), u(k)), \quad x(0) = x_0, \quad k \geq 0 \tag{1.12}$$

where $(x,u) \in \mathbb{X} \times \mathbb{U}$. We use the following notation, for each $x \in \mathbb{X}$ the subset $\mathbb{U}(x) \subset \mathbb{U}$ denotes those controls such that $f(x,u) \in \mathbb{X}$. We assume that $\mathbb{U}(x) \neq \varnothing \quad \forall x \in \mathbb{X}$, thus the system is assumed to be controlled invariant. We also assume that $f(0,0) = 0$. The total cost associated with a given input sequence is defined by

$$V(x_0) = \sum_{k=0}^{\infty} l(x_u(k), u(k)))$$

were the step cost $l : \mathbb{X} \times \mathbb{U} \to \mathbb{R}$ is positive definite, i.e. $l(0,0) = 0$ and $l(x,u) > 0$ if $(x,u) \neq 0$. The optimal value function is defined by

$$V^*(x_0) = \min_{u()} V(x_0)$$

In discrete time the optimal value function can be characterized as the solution to Bellman's equation

$$V^*(x) = \min_{u \in \mathbb{U}(x)} \{V^*(f(x,u)) + l(x,u)\} \tag{1.13}$$

If we know V^*, the optimal feedback controller is given by

$$\mu^*(x) = \text{argmin}_{u \in \mathbb{U}(x)} \{V^*(f(x,u)) + l(x,u)\}$$

Solving Bellman's Equation Value iteration derives from Bellman's famous principle of optimality. Consider the cost of controlling system (1.12) in a finite number of, say N, steps. Doing so in an optimal way would result in a cost

$$V_N^*(x_0) = \min_{u()} \sum_{k=0}^{N-1} l(x_u(k), u(k)) \tag{1.14}$$

This is the same as

$$V_N^*(x) = \min_{u \in \mathbb{U}(x)} \{V_{N-1}^*(f(x,u)) + l(x,u)\} \tag{1.15}$$

Together with initial function $V_0 = 0$, this iterative functional equation defines value iteration. Under suitable conditions the limit $\lim_{N\to\infty} V_N(x)$ exists and coincides with $V^*(x)$. In practice the iteration must, of course, be terminated after a finite number of iterations. Usually one then approximates the optimal infinite horizon controller with the time invariant controller

$$\mu_N^*(x) = \text{argmin}_{u \in \mathbb{U}(x)} \{V_{N-1}^*(f(x,u)) + l(x,u)\}$$

and then uses this to control the system indefinitely. It should be noted that $\mu_N^*(x)$ is *not* the optimal controller for the finite horizon problem 1.14. The N-step optimal controller is given by

$$\mu_j^*(x), \quad 1 \le j \le N$$

which is a time-varying controller. The strategy of using μ_N^* as an approximation to the infinite horizon problem is precisely that used in Receding Horizon Control (RHC) or in Model Predictive Control (MPC). The total cost of using this approximation is

$$V_{\mu_N^*}(x) = \sum_{k=0}^{\infty} l(x_{\mu_N^*}(k), \mu_N^*(x_{\mu_N^*}(k)))$$

We directly see that

$$V_N^*(x) \le V^*(x) \le V_{\mu_N^*}(x)$$

Thus, it is in principle possible to check convergence using this simple inequality. For a recent account on the convergence problem associated with RHC, see [Grüne and Rantzer, 2006].

Exact value iteration, however, is not easy to perform in practice. It is almost always necessary to make approximations. One particular way of formulating an approximation algorithm is presented below.

Relaxed Value Iteration The idea of Relaxed Value Iteration is due to the authors of [Lincoln, 2003; Lincoln and Rantzer, 2006]. See also [Rantzer, 2006] for applications to switching problems.

The following two statements are slight reformulations from [Lincoln, 2003]. Let V_N^* be the N-step optimal cost function obtained by using exact value iteration (1.15). Suppose that $V_N : \mathbb{X} \to \mathbb{R}$ satisfies the following inequalities

$$\begin{aligned} \min_{u \in \mathbb{U}(x)} \{V_{N-1}(f(x,u)) + \beta l(x,u)\} &\leq V_N(x) \\ V_N(x) &\leq \min_{u \in \mathbb{U}(x)} \{V_{N-1}(f(x,u)) + \alpha l(x,u)\} \end{aligned} \tag{1.16}$$

Where $\beta \leq 1 \leq \alpha \in \mathbb{R}$.

PROPOSITION 1.1
Suppose that $V_0 = V_0^*$, then

$$\beta V_N^* \leq V_N \leq \alpha V_N^*, \quad \forall N \in \mathbb{N} \tag{1.17}$$

□

We call the iteration (1.16) relaxed value iteration. It turns out that for some problems it is much easier to find a sequence $\{V_N\}$ that satisfies the inequalities (1.16), compared to the exact iteration. The inequality form also has several other useful properties. The relative bounds obtained can also be used to quantify computation errors made when the exact solution is sought.

The following result can be used to verify that a function V is close to the stationary optimal cost function

PROPOSITION 1.2
Let $\tilde{\mathbb{X}} \subset \mathbb{X}$ with $0 \in \tilde{\mathbb{X}}$ be any invariant subset. If $V \geq 0$ satisfies

$$\begin{aligned} \min_{u \in \mathbb{U}(x)} \{V(f(x,u)) + \beta l(x,u)\} &\leq V(x) \\ V(x) &\leq \min_{u \in \mathbb{U}(x)} \{V(f(x,u)) + \alpha l(x,u)\} \end{aligned} \tag{1.18}$$

Where $\beta \leq 1 \leq \alpha \in \mathbb{R}$, then

$$\beta V^* \leq V \leq \alpha V^*, \quad \forall x \in \tilde{\mathbb{X}} \tag{1.19}$$

□

1.3 Linear Programming and Extreme Points

One of the algorithms in Chapter 4, on constrained control of linear systems, requires enumeration of the extreme points of a certain polyhedron. This section shows how that can be achieved.

An extreme point x of a convex set S is a corner point of S. In other words, if x_1, x_2 are any other points in S, then x is an extreme point of S if $x = \frac{x_1+x_2}{2}$ implies that $x = x_1 = x_2$.

Consider the linear programming problem

$$\begin{aligned} &\min \quad c^T x, \\ &\text{s.t.} \quad Ax = b, \quad x \geq 0. \end{aligned}$$

where $c, x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. The feasible set to this linear program is given by the polyhedron

$$\mathbb{P} = \{x : x \in \mathbb{R}^n, \quad Ax = b, \quad x \geq 0\} \tag{1.20}$$

We assume that $\mathbb{P}$ is non-empty and that the constraint matrix has linearly independent rows, $\text{rank}(A) = m \leq n$. The following two propositions, which can be found in e.g. [Luenberger, 1984], are fundamental results in the theory of linear programming

PROPOSITION 1.3
$x \in \mathbb{R}^n$ is an extreme point of $\mathbb{P}$ if and only if there is a permutation matrix H, note that $HH^T = I$, and an invertible $m \times m$-matrix M such that

$$AH = [M \quad N] \tag{1.21}$$

and

$$H^T x = \begin{bmatrix} x_M \\ x_N \end{bmatrix} = \begin{bmatrix} M^{-1}b \\ 0 \end{bmatrix} \geq 0 \tag{1.22}$$

□

PROPOSITION 1.4
If the primal linear programming problem has a finite optimal solution it is attained at an extreme point of $\mathbb{P}$. □

These two results show that we can solve a linear program by simply enumerating all extreme points of $\mathbb{P}$. The extreme points of $\mathbb{P}$ can be computed as follows: Let

$$y = [A_{k_1}, \quad \ldots, \quad A_{k_m}]^{-1} b$$

where the A_{k_j}'s are linearly independent columns in A, if $y \geq 0$ then x defined by

$$x_{k_j} = \begin{cases} y_j & \text{if } 1 \leq j \leq m \\ 0 & \text{else} \end{cases}$$

is an extreme point of $\mathbb{P}$. The number of extreme points in $\mathbb{P}$ is bounded from above by

$$\left(\begin{array}{c} n \\ m \end{array} \right) = \frac{n!}{(n-m)!m!}$$

which is equal to the number of matrices on the form $[A_{k_1}, \quad \ldots, \quad A_{k_m}]$. Since this number grows very fast *jointly* in (n, m) it is usually inefficient to solve linear programs by enumeration as compared to, for example, the simplex method.

However, observe that for fixed m the number of extreme points in $\mathbb{P}$ grows as a polynomial in n, in fact the asymptotic growth is $O(n^m)$. We shall return to that observation in Chapter 4.

1.4 Positive Polynomials

Consider a polynomial p in variables $x \in \mathbb{R}^n$. Suppose that each coefficient in p depends on a parameter vector via an affine function, i.e. if p_j is the j'th coefficient of p and $t \in \mathbb{R}^w$ is a w-dimensional parameter vector we have the relation

$$t \mapsto p_j(t) = c_0 + \sum_{j=1}^{w} c_j t_j \tag{1.23}$$

where $c_0 ... c_w$ are fixed. Let us denote this parametrized family of polynomials by $p(x;t)$. The goal of this section is to explain how a positivity constraint on $p(x;t)$ can be cast as a linear matrix inequality. The presentation will be brief, for a more thorough discussion see [Parrilo, 2003; Prajna *et al.*, 2002].

$\mathbb{R}[x]$ is the vector space of polynomials in variables $x \in \mathbb{R}^n$. By $\mathbb{R}_d[x]$ we denote the subspace of polynomials of degree at most d. We write $Z_d(x)$ for the column vector consisting of the elements of the canonical basis for $\mathbb{R}_d[x]$, i.e.

$$Z_d(x) = [1 \; x_1 \; x_2 \; .. \; x_n \; x_1 x_2 \; .. \; x_n^d]^T$$

A simple observation is that if $p \in \mathbb{R}_{2d}[x]$ is a sum of squares $p = \sum_{k=1}^{m} p_k^2$ for some $p_k \in \mathbb{R}_d[x]$ then $p \geq 0$ for all $x \in \mathbb{R}^n$. We denote the set of all sum of squares of polynomials by $\Sigma[x]$. The following proposition characterizes all such polynomials

PROPOSITION 1.1
$p \in \Sigma_{2d}[x]$ if and only if

$$p = Z_d(x)^T Q Z_d(x) \tag{1.24}$$

for some positive semidefinite matrix Q. □

This result is important since it allows us to check in a simple way if a given polynomial is a sum of squares. Given a polynomial p, checking if p is a sum of squares can be achieved using semidefinite programming as follows: First identify coefficients in (1.24), this gives an affine constraint on the elements of Q, then by taking the intersection with the convex cone of positive semidefinite matrices results in a convex constraint.

EXAMPLE 1.2
Consider the following polynomial

$$p = x_1^2 x_2^2 x_3^2 - 2x_1^2 x_2 x_3 - 2x_1 x_2 x_3 + x_1^2 + 2x_1 + 2$$

To verify that p is a sum of squares we let

$$Z = [1 \; x_1 \; x_1 x_2 x_3]^T, \quad Q = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & q_4 & q_5 \\ q_3 & q_5 & q_6 \end{bmatrix}$$

If we equate terms in p and $Z^T Q Z$ we get

$$Q = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & q_4 & q_5 \\ q_3 & q_5 & q_6 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

Since Q can be factorized as

$$Q = \begin{bmatrix} -1 & 1 \\ -1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

we have $Q \succeq 0$ and p is a sum of squares

$$p = (x_1 x_2 x_3 - x_1 - 1)^2 + 1$$

Not only have we proved that p is non-negative we have in fact showed that $p \geq 1$ and moreover this bound is optimal since $p(-1, 0, 0) = 1$. □

It is equally easy to check if a parametrized, as in (1.23), polynomial is a sum of squares by identifying the coefficients in

$$p(x; t) = Z_d(x)^T Q Z_d(x) \tag{1.25}$$

If we define $q = \text{vec}(Q)$, so that the columns in Q are stacked in q, the constraint (1.25) can be written as

$$A[t^T \quad q^T]^T = b \tag{1.26}$$

for appropriate constants A and b.

The other implication is false, i.e. even if a polynomial p is non-negative p is not necessarily a sum of squares, see [Parrilo, 2003]. Thus the above procedure gives sufficient conditions for positivity on $\mathbb{R}^n$. This fact shows that checking global positivity of a polynomial using the outlined method can be somewhat conservative.

In this thesis we focus on positivity on compact sets, and this case is less conservative. Consider a set

$$\mathbb{X} = \{x : h_k(x) \geq 0, k = 1..m\} \tag{1.27}$$

with $h_k \in \mathbb{R}[x]$. We associate with $\mathbb{X}$ a set of polynomials

$$G_{\mathbb{X}} = \{p : p = \sigma_0 + \sum_{k=1}^{m} \sigma_k h_k, \quad \sigma_k \in \Sigma[x]\} \tag{1.28}$$

Similar to the global case we clearly have

LEMMA 1.1
If $p \in G_{\mathbb{X}}$ then $p \geq 0$ on $\mathbb{X}$.

□

The following remarkable partial converse, see [Putinar, 1993], will be useful

THEOREM 1.3—PUTINAR
Let $\mathbb{X}$ be as in (1.27). Suppose that there is a real number r such that $r^2 - \sum_{k=1}^{n} x_k^2 \in G_{\mathbb{X}}$, then $p \in \mathbb{R}[x]$ is positive on $\mathbb{X}$ only if $p \in G_{\mathbb{X}}$.

□

EXAMPLE 1.3
The following polynomial

$$p = -x_1^2x_2^2 - 2x_1^2x_2 - 2x_1x_2 + 2x_1^2 + 2x_1 + 1 - x_1^4 + x_2^2 - x_2^4$$

is not non-negative on $\mathbb{R}^2$, e.g $p(1,1) = -1$. It is, however, non-negative on the unit disc

$$\mathbb{X} = \{x : h_1 = 1 - x_1^2 - x_2^2 \geq 0\}$$

To show that we can use the outlined procedure to arrive at the following identity

$$p = \sigma_0 + \sigma_1 h_1 = (x_1x_2 - x_1 - 1)^2 + (x_1^2 + x_2^2)(1 - x_1^2 - x_2^2)$$

□

Linear Programming Alternative

We can combine Theorem 1.3 and Proposition 1.1 to optimize polynomials over compact semialgebraic sets. Doing so requires us to solve convex optimization problems with semidefinite constraints. These problems grow fast in size as the degree of the polynomials grows. Current implementations of semidefinite program solvers can solve medium-sized problems very efficiently. Large-scale linear programming problems, however, can be solved much more efficiently as compared to semidefinite programming problems. Therefore, when the degrees of the involved polynomials grow large, it would be advantageous to be able to cast polynomial positivity constraints as linear inequalities, and thus be able to use linear programming codes to verify positivity. This can in fact be achieved by using the following theorem of Handelman [Prestel and Delzell, 2001]

THEOREM 1.4—HANDELMAN
Let $\mathbb{X}$ be non-empty, and defined as in (1.27). Suppose that the h_k's contain the subsequence $h_1 = b_1 - a_1^Tx, ..., h_q = b_q - a_q^Tx$ such that the set

$$\{x : h_k(x) \geq 0, k = 1..q\}$$

is compact. Then $p \in \mathbb{R}[x]$ is positive on $\mathbb{X}$ only if p can be written as

$$p = \sum_{0 \leq \nu \in \mathbb{N}^q} c_\nu (b - a^Tx)^\nu \tag{1.29}$$

for some finite number of non-negative coefficients c_ν.

□

Here we use the notation

$$(b - a^T x)^\nu = \prod_{k=1}^{q} (b_k - a_k^T x)^{\nu_k}$$

Of course, just as in Lemma 1.1 if p can be written as in equation (1.29) this implies that $p \geq 0$ on $\mathbb{X}$.

We apply this theorem in similar fashion as in the application of Theorem 1.3. Let $p(x;t)$ be a parametrized in t polynomial, with coefficients as in equation (1.23), put

$$p(x;t) = \sum_{\nu \in D} c_\nu (b - a^T x)^\nu \tag{1.30}$$

where D is a finite set of vectors in $\mathbb{N}^q$. Identification of coefficients on both sides gives a linear equality constraint on the variables $t_1 ... t_w$ and $\nu \in D$. Adding to this constraint a linear objective function results in a linear programming problem.

See [Lasserre, 2002] for a comparison of linear programming versus semidefinite programming relaxation techniques for polynomial optimization.

2

Approximate Policy Iteration

2.1 Introduction

This chapter is devoted to the development of an approximate policy iteration algorithm, the purpose of the algorithm being to compute approximate solutions to the stationary HJB-equation. The main idea is to replace an equality constraint with two inequalities. The novel part of the main result is to show precisely how to perturb the exact equation to get a monotonically converging algorithm. A convergence proof is given, including the rate of convergence. We apply the continuous time result to problems with polynomial dynamics and quadratic penalty on the control variable. The resulting inequalities are verified using polynomial relaxation techniques.

Policy iteration or successive approximation in policy space and other similar methods, were already discussed by Bellman in [Bellman, 1957]. A theoretical analysis of policy iteration in continuous time is given in [Leake and Liu, 1967]. They show how the solution of the HJB-equation can be reduced to the solution of a sequence of first order linear partial differential equations. However, they do not consider any computations. Related to this is [Kleinman, 1968], where the author presents an algorithm to solve the algebraic Riccati equation. More recently, the authors of [Beard *et al.*, 1998] apply the Galerkin spectral method to obtain solutions to the aforementioned sequence of linear partial differential equations. However, no bounds on the approximations are given and assumptions which are difficult to check must be fulfilled. Moreover, the main computational task in the algorithm proposed by [Beard *et al.*, 1998] is multidimensional integration, such computations become prohibitive for systems with more then a few states. On the other hand, an advantage of that method is that it can be applied to problems where the system dynamics are not necessarily modeled with polynomials. This is in contrast to the computational

method presented in this chapter.

There are several other relevant methods that can be used to compute an approximate solution to the HJB-equation. For example, the work in [Garrard, 1969; Nishikawa and Itakura, 1971] where the authors use various power series expansion strategies, with various assumptions, to obtain approximate solutions to the HJB-equation. These methods can sometimes be used to compute acceptable local estimates, using only a few terms. Although higher order approximations are possible to compute, the complexity is often prohibitive.

Motivated by LQ-control, another approach is to write the nonlinear system in a linear like representation and derive a state dependent Riccati equation, see [Huang and Lu, 1996]. This approach is taken in [Prajna *et al.*, 2004], where the authors use representations of positive polynomials to derive sufficient conditions for upper bounds on the value function.

In [Markman and Katz, 2000] the authors use a discretization and interpolation technique. The state space is discretized and the open loop minimum control is computed for each point. These are then combined to form a feedback controller. The drawback is that gridding techniques are expensive in that such methods require computations that scale exponentially in state dimension.

2.2 Continuous Time Version

There are several references, see [Bardi and Capuzzo-Dolcetta, 1997; Fleming and Soner, 1993], in which optimal control theory with weak interpretations of derivatives is developed. The reason for this is that for many optimal control problems in continuous time the optimal value function does not have a classical gradient. In the work presented in this chapter we do not assume that the optimal value function is differentiable, but instead we consider a differentiable sequence of functions that can be shown to converge monotonically to the optimal cost.

Problem Setup

Consider a continuous time system

$$\dot{x} = f(x,u) \tag{2.1}$$

with $(x,u) \in \mathbb{X} \times \mathbb{U} \subset \mathbb{R}^n \times \mathbb{R}^m$. The initial state $x_0 := x(0) \in \mathbb{X}$ is given. We denote the instantaneous cost function by $l(x,u)$. The origin is a fixed point of system (2.1), so $0 = f(0,0)$. The instantaneous cost function l is continuous positive definite, in other words $l(x,u) > 0$ for all $(x,u) \neq (0,0)$

and $l(0,0) = 0$. The total cost is defined as

$$V = \int_0^\infty l(x,u)dt \tag{2.2}$$

The optimal control problem we consider is defined by

$$V^*(x_0) = \inf_{u(\cdot)} V \quad \text{such that (2.1) is satisfied.} \tag{2.3}$$

Exact Policy Iteration

Suppose that the cost function V_{μ_j} given by (2.2) is finite on $\mathbb{X}$ when using the feedback controller $u(t) = \mu_j(x(t))$. Assume also that V_{μ_j} is differentiable. The cost can be computed by solving

$$DV_{\mu_j} \cdot f(x, \mu_j(x)) + l(x, \mu_j(x)) = 0 \tag{2.4}$$

Define a new controller by

$$\mu_{j+1}(x) = \text{argmin}_u \{DV_{\mu_j} \cdot f(x,u) + l(x,u)\} \tag{2.5}$$

The repetition of these two steps constitutes the so called exact policy iteration. Then, it is not hard to show that

$$V_{\mu_j} \geq V_{\mu_{j+1}} \tag{2.6}$$

i.e. we have monotonic convergence, this fact follows as a special case from the main result below.

There are at least two problems with this iteration. The cost V_{μ_j} might not exist as a differentiable function. Even if it is differentiable, it is generally impossible to find the exact solution to (2.4). If approximate solutions are used convergence may be lost. It is necessary that computational methods keep track of the successive errors. In the next section such an approximate iteration is proposed.

Approximate Policy Iteration

In the following section, differentiable functions that approximate the true cost functions V_{μ_j} are denoted by V_j. Note that we do not assume that the true costs V_{μ_j} are differentiable.

Assume that there is a feedback controller μ_0 and a differentiable function V_0 such that

$$0 \geq DV_0 \cdot (f(x, \mu_0(x)) + l(x, \mu_0(x)), \quad \forall x \in \mathbb{X}$$

Define

$$T_j(x) = -DV_{j-1} \cdot f(x, \mu_j(x)) - l(x, \mu_j(x))$$

THEOREM 2.1—APPROXIMATE POLICY ITERATION
Suppose that the sequence $\{(\alpha_j, \mu_j, V_j)\}_{j\geq 1}$ with V_j positive definite for $j \geq 1$ satisfies

$$0 \geq DV_j \cdot f(x, \mu_j(x)) + l(x, \mu_j(x)), \tag{2.7}$$
$$0 \leq DV_j \cdot f(x, \mu_j(x)) + l(x, \mu_j(x)) + \alpha T_j(x) \tag{2.8}$$
$$0 \leq T_j(x) \tag{2.9}$$
$$0 \leq \alpha_j \leq 1 \tag{2.10}$$

and that no solution x_{μ_j} leaves $\mathbb{X}$. Then for every $j \geq 1$ it holds

$$V_{j-1} \geq V_j \geq V_{\mu_j}$$

□

REMARK 2.1
Note that the condition $T_j(x) \geq 0$ is redundant if $\alpha_j > 0$. □

PROOF 2.1
Let $x_{\mu_j}(t)$ denote the trajectory as a result of applying μ_j and consider inequality (2.7)

$$\begin{aligned}
&V_j(x_{\mu_j}(0)) - V_j(x_{\mu_j}(T)) \\
&= \int_0^T -DV_j(x_{\mu_j}(t)) \cdot f(x_{\mu_j}(t), \mu_j(x_{\mu_j}(t)))dt \\
&\geq \int_0^T l(x_{\mu_j}(t), \mu_j(x_{\mu_j}(t)))dt
\end{aligned}$$

Thus

$$V_j(x_{\mu_j}(0)) \geq V_j(x_{\mu_j}(T)) + \int_0^T l(x_{\mu_j}(t), \mu_j(x_{\mu_j}(t)))dt$$

Since V_j and l are continuous and positive definite it follows that $x_{\mu_j}(t) \to 0$ as $t \to \infty$ and therefore $V_j \geq V_{\mu_j}$. Also, by inequality (2.8)

$$\begin{aligned}
0 &\leq DV_j \cdot f(x, \mu_j(x)) + l(x, \mu_j(x)) \\
&\quad + \alpha_j(-DV_{j-1} \cdot f(x, \mu_j(x)) - l(x, \mu_j(x))) \\
&= DV_j \cdot f(x, \mu_j(x)) - DV_{j-1} \cdot f(x, \mu_j(x)) \\
&\quad - (1 - \alpha_j)(-DV_{j-1} \cdot f(x, \mu_j(x)) - l(x, \mu_j(x))) \\
&= DV_j \cdot f(x, \mu_j(x)) - DV_{j-1} \cdot f(x, \mu_j(x)) \\
&\quad - (1 - \alpha_j)T_j(x) \\
&\leq D(V_j - V_{j-1}) \cdot f(x, \mu_j(x))
\end{aligned}$$

the last inequality implies that

$$\begin{aligned} & V_{j-1}(x_{\mu_j}(0)) - V_{j-1}(x_{\mu_j}(T)) \\ &= \int_0^T (-DV_{j-1} \cdot f(x, \mu_j(x)))dt \\ &\geq \int_0^T (-DV_j \cdot f(x, \mu_j(x)))dt \\ &= V_j(x_{\mu_j}(0)) - V_j(x_{\mu_j}(T)) \end{aligned}$$

by sending $T \to \infty$ we conclude $V_{j-1} \geq V_j$. □

The result shows that V_j is bounded from below, for by definition $V_{\mu_j} \geq V^*$. Moreover $\{V_j\}_{j\geq 0}$ is monotonically non-increasing. To prove global convergence it is necessary to impose, at least, one more condition on the sequence $\{V_j, \mu_j, \alpha_j\}$. In the next result we provide such a condition

THEOREM 2.2—GLOBAL CONVERGENCE
Select $\{\mu_j\}_{j\geq 0}$ according to

$$\mu_{j+1}(x) = \arg\min_u \{DV_j \cdot f(x,u) + l(x,u)\} \tag{2.11}$$

suppose that $\{V_j\}_{j\geq 1}$ satisfies inequalities (2.7)-(2.8) and in addition

$$\alpha_j < 1$$

Then, for any $x \in \mathbb{X}$

$$V_j(x) \to V^*(x) \tag{2.12}$$

□

PROOF 2.2
$V_{j-1} \geq V_j$ and the fact that V_j is bounded from below $V_j \geq V^*$ shows that there must be a limit $V_j(x) \to \hat{V}(x)$. We shall show that our choice (2.11) implies that $\hat{V}(x) = V^*(x)$.

Unless there is a point x such that $V_{j-1}(x) > V_j(x)$, in which case the sequence $\{V_j\}$ is strictly improving, we have $V_{j-1}(x) = V_j(x)$. Now, by using the inequalities (2.7)-(2.8) and the fact that $\alpha_j < 1$ we have $T_j = 0$, therefore

$$\begin{aligned} 0 &= T_j \\ &= -DV_{j-1} \cdot f(x, \mu_j) - l(x, \mu_j(x)) \\ &= -\min_u \{DV_{j-1} \cdot f(x,u) + l(x,u)\} \end{aligned}$$

□

How To Find an Initializer?

To use Theorem 2.1 in an algorithm one must find an initializer μ_0 and a corresponding V_0 such that (2.2) is satisfied. Although any locally stabilizing μ_0 could be used for this purpose, it is in general difficult to find such a μ_0 for a non-linear system. We will outline how a computational procedure could be constructed to simplify the initialization.

We assume that there is a constant $C > 0$ such that

$$l(x,u) \geq C(|x|^2 + |u|^2)$$

and that the minimum in (2.3) is attained at μ^* and that the optimal trajectory $(x^*(t), \mu^*(x^*(t)))$ is a well defined, piecewise continuous, solution to (2.1). As a consequence of these assumptions, the optimal cost function V^* is positive definite and continuous on its domain, i.e. at points in

$$\Sigma_\infty^* = \{x : x \in \mathbb{R}^n, V^*(x) < \infty\}$$

Moreover, each sub-level set

$$\Sigma_\rho^* = \{x : x \in \mathbb{R}^n, V^*(x) \leq \rho\}$$

is a connected compact subset of Σ_∞^*, thus $\cup_\rho \Sigma_\rho^* = \Sigma_\infty^*$ For each $T > 0$ and $x(0) \in \Sigma_\infty^*$ we have

$$\rho_0 = V^*(x(0)) = V^*(x^*(T)) + \int_0^T l(x^*(t), \mu^*(x^*(t)))dt > V^*(x(T)) \quad (2.13)$$

thus $x(T) \in \Sigma_{\rho_0}^*$. It follows that for any $\rho \geq 0$, the set Σ_ρ^* is forward invariant for each solution $x^*(t)$ to $\dot{x} = f(x^*, \mu^*(x))$.

Consider now the case when the system dynamics (2.1) and instantaneous cost are twice continuously differentiable, define

$$\begin{aligned} A &= f_x(0,0) \\ B &= f_u(0,0) \\ Q &= l_{xx}(0,0) \\ R &= l_{uu}(0,0) \end{aligned}$$

Suppose that (A,B) is stabilizable and that $Q > 0, R > 0$. Consider the LQ-problem for the linearized system

$$\begin{aligned} &V_{LQ} = \min_u \quad \int_0^\infty x^T Q x + u^T R u dt, \\ &\text{such that} \quad \dot{x} = Ax + Bu \end{aligned}$$

Let $V_{LQ} = x^T P x$ be the optimal cost and let $\mu_{LQ} = L^T x$ be the corresponding optimal feedback controller. It has been shown in [Lukes, 1969] that the controller μ_{LQ} stabilizes the nonlinear system (2.1) on some set $\mathbb{W}$ that contains the origin its interior and for each $x \in W$

$$V^*(x) = V_{LQ} + o(|x|^2)$$
$$\mu^*(x) = \mu_{LQ} + o(|x|)$$

We can choose $\mathbb{W} = \Sigma^*_\rho$ for some $\rho > 0$. Consider now a sequence $\{(\mu_j, V_j)\}_{j\geq 1}$ that satisfies Theorem (2.1), and μ_j continuous, selected as in(2.11). The V_j's are assumed differentiable across the boundary of the invariance region Σ^*_ρ. Due to compactness, $\mu_j \to \mu^*$ uniformly on Σ^*_ρ. Every optimal trajectory starting outside of Σ^*_ρ points into Σ^*_ρ at its boundary. With these facts one could hope that also each x_{μ_j} would point into Σ^*_ρ, at least in the limit, and with this find a larger invariance region $\Sigma^*_{\hat{\rho}} \supset \Sigma^*_\rho$. We could repeat the procedure to obtain a sub sequence $V_{\mu_{j_k}}$ valid on growing nested sequence $\Sigma^*_{\rho_k}$ and conclude that $\mu_j \to \mu^*$ on Σ^*_ρ for all ρ. However, the outlined construction would require several additional regularity assumptions.

2.3 Application on Input-Affine Systems

We would like to deduce an algorithm from the results in the previous section. These results are valid for general continuous systems. However, from a computational point of view, it is too difficult in general to do the pointwise minimization in (2.11). We therefore consider input-affine systems

$$\dot{x} = f(x) + g(x)u \tag{2.14}$$

and we take the instantaneous cost to be quadratic in the control variable

$$l(x,u) = q(x) + u^T R u$$

with $q(x) > 0$ if $x \neq 0$ and $q(0) = 0$, also $R > 0$. With this choice there is a unique minimizer in (2.11), given by

$$\begin{aligned}\mu_{j+1}(x) &= \operatorname{argmin}_u \{DV_j \cdot (f(x) + g(x)u) + q(x) + u^T R u\} \\ &= -\frac{1}{2} R^{-1} g(x)^T DV_j\end{aligned}$$

Denote by $P(\mu_j)$ the following optimization program:

$$
\begin{aligned}
&P(\mu_j) \quad \mapsto \min \alpha_j, \\
&\text{such that} \quad V_j(0) = 0 \quad \text{and } \forall x \in \mathbb{X} \\
&0 \geq DV_j \cdot (f(x) + g(x)\mu_j(x)) + l(x, \mu_j(x)) \\
&0 \leq DV_j \cdot (f(x) + g(x)\mu_j(x)) + l(x, \mu_j(x)) + \alpha_j T_j(x) \\
&V_j \geq 0
\end{aligned}
$$

We would like to be able to solve $P(\mu_j)$ repeatedly, and to do that we need more structure. First, the exact cost functions V_{μ_j} belong, in general, to an infinite dimensional space. To be able to do computations it is necessary to restrict the search for an approximation V_j to a subspace $H_{d_j} \subset C^1(\mathbb{X}, \mathbb{R})$ of dimension d_j. Moreover, the inequality constraints appearing in $P(\mu_j)$ are intractable to verify in general. To get something that is tractable we consider systems (2.14) that are modeled by polynomials, $f, g \in \mathbb{R}[x]$, also we take $H_{d_j} \subset \mathbb{R}[x]$. For notational simplicity, let us assume that $\mathbb{X}$ can be described with only one polynomial, for example a closed ball centered at the origin with radius a, i.e. $\mathbb{X} = \{x : a^2 - |x|^2 \geq 0\}$. Moreover, let $s_L, s_U, s_V \in \Sigma_m[x]$. We now specialize $P(\mu_j)$ to:

$$
\begin{aligned}
&P(\mu_j, d_j) \quad \mapsto \min \alpha_j, \\
&\text{such that} \\
&\quad -(DV_j \cdot (f(x) + g(x)\mu_j(x)) + l(x, \mu_j(x))) - s_L(a^2 - |x|^2) \in \Sigma[x] \\
&\quad (DV_j \cdot (f(x) + g(x)\mu_j(x)) + l(x, \mu_j(x)) + \alpha_j T_j(x)) - s_U(a^2 - |x|^2) \in \Sigma[x] \\
&\quad V_j - s_V(a^2 - |x|^2) \in \Sigma[x] \\
&\quad s_L, s_u, s_V \in \Sigma_m[x] \\
&\quad V_j(0) = 0
\end{aligned}
$$

By the discussion in Chapter 1 this is a semidefinite program, which can be solved efficiently as long as the dimension of H_{d_j} is not too high.

We summarize the discussion in the following pseudo code

ALGORITHM 2.1—APPROXIMATE POLICY ITERATION

Let μ_0 be any locally stabilizing feedback controller.
repeat
  repeat
    Solve $P(\mu_j, d_j)$ for V_j.
    if $P(\mu_j, d_j)$ is not feasible **then**
      increase d_j
    end if
  until $P(\mu_j, d_j)$ is feasible
  $\mu_{j+1} = -\frac{1}{2}R^{-1}g(x)^T DV_j$
until "convergence"

□

Examples

In this section we give two examples of the proposed method. The first example is simple in the sense that we can obtain an analytic solution to the HJB-equation for comparison. The second example is adopted from [Beeler *et al.*, 2000], which in turn originates from [Garrard and Jordan, 1977]. In [Beeler *et al.*, 2000] the authors compare some of the existing methods for computation of suboptimal controllers to nonlinear systems.

Both examples were implemented in Yalmip [Löfberg, 2004], with Sedumi [Sturm, 1999] as the underlying semidefinite programming solver.

EXAMPLE 2.1

Consider the one dimensional linear system

$$\dot{x} = -x + u \tag{2.15}$$

with instantaneous cost $l(x,u) = x^2 + x^4 + u^2$, note that $q(x) = x^2 + x^4$ is not quadratic in the state. The exact solution to this problem was computed in Chapter 1. The optimal value function is

$$V^*(x) = -x^2 + 2\frac{(2+x^2)^{3/2} - 2\sqrt{2}}{3} \tag{2.16}$$

and the optimal feedback controller is

$$\mu^*(x) = x - x\sqrt{2+x^2} \tag{2.17}$$

We took $\mu_0 = 0$ as the initial controller and $\mathbb{X} = [-1, 1]$ as the approximation region. Approximations to the value function with degree 4 and 6

were computed

$$\begin{aligned} V^4 &= 0.4176x^2 + 0.1650x^4 \\ V^6 &= 0.4149x^2 + 0.1739x^4 - 0.01030x^6; \end{aligned} \tag{2.18}$$

with associated controls

$$\begin{aligned} \mu^4 &= -0.4176x - 0.3300x^3 \\ \mu^6 &= -0.4149x - 0.3478x^3 + 0.03090x^5 \end{aligned} \tag{2.19}$$

For both degree 4 and 6 the main loop in Algorithm 2.1 was successfully executed four times, but at the fifth iteration problem $P(\mu_j, d_j)$ became infeasible. Recall that at iteration j the α_j-parameter can be interpreted as a measure of how good V_j approximates V_{μ_j}. The α-parameter versus iteration is shown in Table 2.1. Consider first the row corresponding to degree 4. That the sequence of α values is monotonically increasing means that is becomes more and more difficult to improve the approximation. At the fifth iteration no more improvements are possible if the degree of V_j is kept fixed. Consider now the second row in Table 2.1, corresponding to a degree 6 approximation. That the α value in each iteration is less than the corresponding value for the degree 4 approximation means that the convergence is faster for the degree 6 approximation, as expected.

To compare the presented algorithm with power series methods, consider the Taylor expansion of the exact solution at the origin

$$V^*(x) = 0.4142\,x^2 + 0.1768\,x^4 - 0.01473\,x^6 + o(x^7)$$

and

$$u^*(x) = -0.41421\,x - 0.3536\,x^3 + 0.04419\,x^5 + o(x^6)$$

The reason for the differences between the computed approximations and the Taylor expansion is that the method presented in this chapter computes a uniform approximation whereas the Taylor expansion is a local approximation. For the sixth degree approximation we have the following estimate

$$\max_{x\in\mathbb{X}} |V^*(x) - V^6(x)| \approx 10^{-2} \max_{x\in\mathbb{X}} |V^*(x) - V_T(x)| \tag{2.20}$$

where V_T is the sixth degree Taylor polynomial. Although this is a simple example, it gives some indication of the usefulness of our method as compared to power series expansion methods.

□

Deg\Iter.	1	2	3	4
4	1e-4	0.03960	0.5547	0.9945
6	1e-4	0.002900	0.04840	0.2968

Table 2.1 α-parameter vs. iteration in example 1.

EXAMPLE 2.2
In this example we consider a flight control problem. The system is modeled with three states x_1 is the angle of attack, x_2 is the flight path angle and x_3 the rate of change of flight path angle. The control variable u is the tail deflection angle. The states and control variable should be interpreted as deviations from some setpoint. The model is as follows

$$\begin{aligned} f_1(x) =& -0.877x_1 + x_3 + 0.47x_1^2 - 0.088x_1x_3 - 0.019x_2^2 \\ &+ 3.846x_1^3 - x_1^2x_3 \\ f_2(x) =& x_3 \\ f_3(x) =& -4.208x_1 - 0.396x_3 - 0.47x_1^2 - 3.564x_1^3 \end{aligned} \tag{2.21}$$

and

$$g^T = [-0.215 \quad 0 \quad -20.967] \tag{2.22}$$

In this example the instantaneous cost is $l(x,u) = 0.25x^Tx + u^2$. Approximations were computed on

$$\mathbb{X} = \{x : -0.1 \leq x_1 \leq 25(\pi/180) + 0.1, \quad x_2^2 + x_3^2 \leq 0.1\}$$

To evaluate the computed controllers we used compared their ability to bring the system to rest after an initial perturbation in the angle of attack, $x_0^T = [25(\pi/180) \quad 0 \quad 0]$. The algorithm was initialized with μ_0 equal to the LQ-controller for the linearized system.

The results are summarized in Figures 2.1-2.3. The discussion about the α-parameter versus iteration in the previous example also applies in this example. In fact, the pattern is even more pronounced in this example, as shown in Figure 2.3.

Moreover, the results can also be compared to the results obtained in [Beeler *et al.*, 2000] for the same problem. The best controller for this problem were obtained with a discretization-interpolation method. The performance of that controller is about the same as that we obtained with the third degree approximation using the method presented in this chapter. The reported computation time required for the discretization-interpolation method is about 6000 seconds, compared to the method in

this chapter which required 60 seconds. Also, our controller is a third degree polynomial, to be compared with the complicated representation for the discretization-interpolation method. Note, the numerical computations in [Beeler *et al.*, 2000] was done on a platform that is similar to the one we used for our computations.

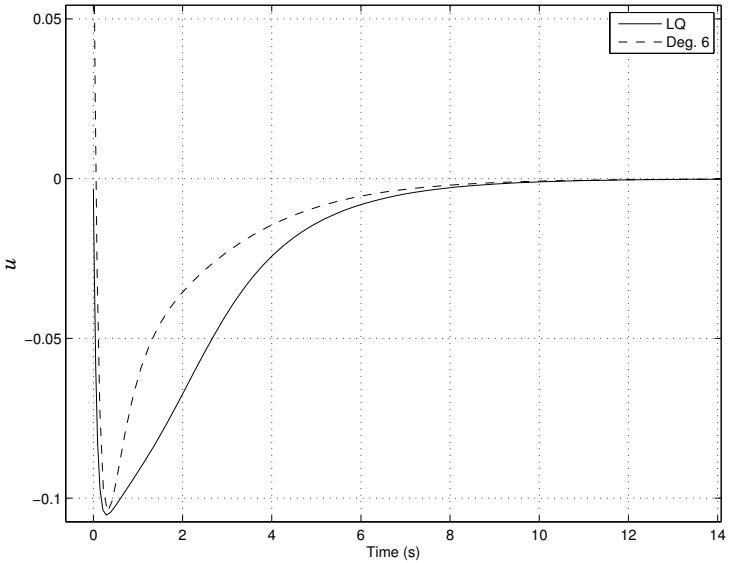


Figure 2.1 Control signal for different degrees of approximation in Example 2.

□

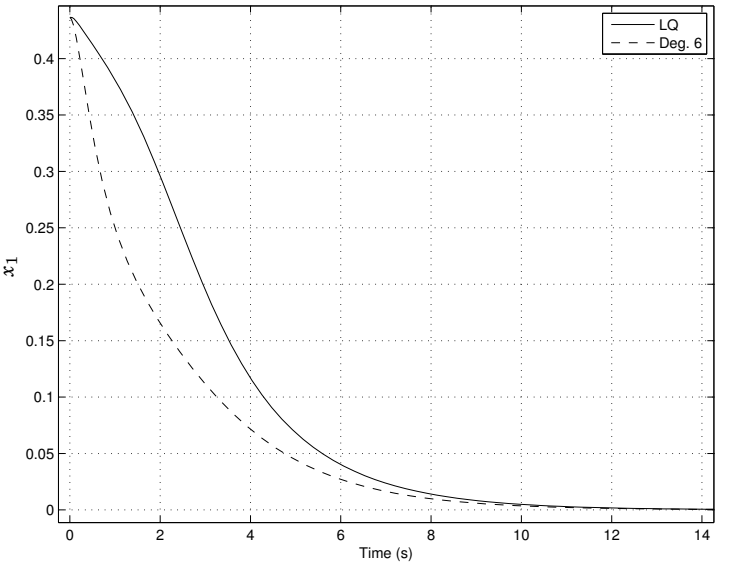


Figure 2.2 State x_1 for different degrees of approximation in Example 2.

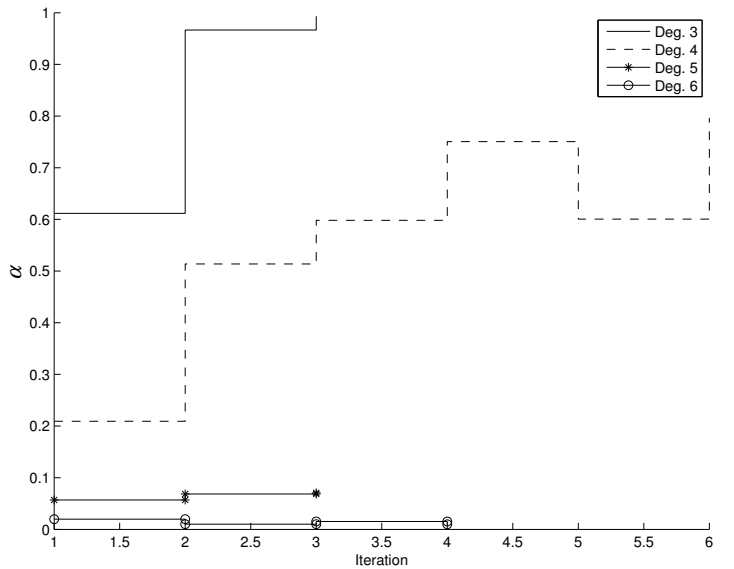


Figure 2.3 α-parameter vs. iteration in Example 2.

2.4 Discrete Time Version with Convergence Rate

The approximate policy iteration is also valid in discrete time. Although the proofs are similar to the ones in continuous time, with integrals substituted for summations, they are given here for completeness. Moreover, the rate of convergence is also given in this section.

Consider a discrete time system

$$x(k+1) = f(x(k), u(k)) \tag{2.23}$$

with $(x(k), u(k)) \in \mathbb{X} \times \mathbb{U} \subset \mathbb{R}^n \times \mathbb{R}^m$. The initial state $x_0 := x(0) \in \mathbb{X}$ is given. We denote the step cost function by $l(x,u)$. The origin is a fixed point of system (2.23), that is $0 = f(0,0)$. The step cost function l is continuous positive definite, in other words $l(x,u) > 0$ for all $(x,u) \neq (0,0)$ and $l(0,0) = 0$. The total cost is defined as

$$V = \sum_{k=0}^{\infty} l(x(k), u(k)) \tag{2.24}$$

The optimal control problem we consider is defined by

$$V^*(x_0) = \inf_{u(\cdot)} V \quad \text{such that (2.23) is satisfied.} \tag{2.25}$$

Suppose that the cost function V_{μ_j} given by (2.24) is finite on $\mathbb{X}$ when using the controller μ_j. The cost can be computed by solving

$$V_j(x) = V_j(f(x, \mu_j(x))) + l(x, \mu_j(x)) \tag{2.26}$$

Define a new controller by

$$\mu_{j+1}(x) = \arg\min_u \{V_j(f(x,u)) + l(x,u)\} \tag{2.27}$$

The repetition of these two steps constitutes the exact policy iteration in discrete time.

Approximate Policy Iteration in Discrete Time

Assume that there is a feedback controller μ_0 and a function V_0 such that

$$V_0(x) \geq V_0(f(x, \mu_0(x)) + l(x, \mu_0(x)), \quad \forall x \in \mathbb{X}$$

Define

$$T_j(x) = V_{j-1}(x) - V_{j-1}(f(x, \mu_j(x)) - l(x, \mu_j(x)).$$

THEOREM 2.3—APPROXIMATE POLICY ITERATION
Suppose that the sequence $\{(\mu_j, V_j)\}_{j\geq 1}$ with V_j positive definite satisfies

$$V_j(x) \geq V_j(f(x,\mu_j(x))) + l(x,\mu_j(x)) \tag{2.28}$$
$$V_j(x) \leq V_j(f(x,\mu_j(x))) + l(x,\mu_j(x)) + \alpha_j T_j(x) \tag{2.29}$$
$$0 \leq T_j(x) \tag{2.30}$$
$$0 \leq \alpha_j \leq 1 \tag{2.31}$$

Then for every $j \geq 1$ it holds

$$V_{j-1} \geq V_j \geq V_{\mu_j}$$

□

REMARK 2.2
Note that the condition $T_j(x) \geq 0$ is redundant if $\alpha_j > 0$. □

PROOF 2.3
Now let $x_{\mu_j}(k)$ denote the trajectory as a result of applying μ_j and consider inequality (2.28)

$$\begin{aligned} &V_j(x_{\mu_j}(0)) - V_j(x_{\mu_j}(t+1)) \\ &= \sum_{k=0}^{t} (V_j(x_{\mu_j}(k)) - V_j(x_{\mu_j}(k+1))) \\ &\geq \sum_{k=0}^{t} l(x_{\mu_j}(k), \mu_j(k)) \end{aligned}$$

Thus

$$V_j(x_{\mu_j}(0)) \geq V_j(x_{\mu_j}(t+1)) + \sum_{k=0}^{t} l(x_{\mu_j}(k), \mu_j(k))$$

Since $V_j \geq 0$ and $l(x,u) > 0$ if $(x,u) \neq 0$ we have $x_{\mu_j}(t) \to 0$ as $t \to \infty$ and therefore $V_j \geq V_{\mu_j}$. Also

$$\begin{aligned} V_j(x) &\leq V_j(f(x,\mu_j(x))) + l(x,\mu_j(x)) \\ &\quad + \alpha(V_{j-1}(x) - V_{j-1}(f(x,\mu_j(x)) - l(x,\mu_j(x))) \\ &= V_j(f(x,\mu_j(x))) + V_{j-1}(x) - V_{j-1}(f(x,\mu_j(x)) \\ &\quad - (1-\alpha)(V_{j-1}(x) - V_{j-1}(f(x,\mu_j(x))) - l(x,\mu_j(x))) \\ &\leq V_j(f(x,\mu_j(x))) + V_{j-1}(x) - V_{j-1}(f(x,\mu_j(x)) \end{aligned}$$

the last inequality implies that

$$\begin{aligned} & V_{j-1}(x_{\mu_j}(0)) - V_{j-1}(x_{\mu_j}(t+1)) \\ &= \sum_{k=0}^{t} (V_{j-1}(x_{\mu_j}(k)) - V_{j-1}(x_{\mu_j}(k+1)) \\ &\geq \sum_{k=0}^{t} (V_j(x_{\mu_j}(k)) - V_j(x_{\mu_j}(k+1)) \\ &= V_j(x_{\mu_j}(0)) - V_j(x_{\mu_j}(t+1)) \end{aligned}$$

by sending $t \to \infty$ we conclude $V_{j-1} \geq V_j$. □

The result shows that V_j is bounded from below, for by definition $V_{\mu_j} \geq V^*$. Moreover $\{V_j\}_{j\geq 0}$ is monotonically non-increasing. To prove global convergence it is necessary to impose, at least, one more condition on the sequence $\{V_j, \mu_j, \alpha_j\}$. In the next result we provide such a condition

THEOREM 2.4
Select $\{\mu_j\}_{j\geq 1}$ according to

$$\mu_{j+1}(x) = \arg\min_u \{V_j(f(x,u)) + l(x,u)\} \tag{2.32}$$

suppose that $\{V_j\}_{j\geq 1}$ satisfies inequalities (2.28)-(2.29) and in addition

$$\alpha_j < 1$$

Then, for any $x \in \mathbb{X}$

$$V_j(x) \to V^*(x)$$

□

PROOF 2.4
$V_{j-1} \geq V_j$ and the fact that V_j is bounded from below $V_j \geq V^*$ shows that there must be a limit $V_j(x) \to \hat{V}(x)$. We shall show that our choice (2.32) implies that $\hat{V}(x) = V^*(x)$.

Unless there is a point x such that $V_{j-1}(x) > V_j(x)$, in which case the sequence $\{V_j\}$ is strictly improving, we have $V_{j-1}(x) = V_j(x)$. But then we can use inequalities (2.28)-(2.29) and the fact that $\alpha_j < 1$ to show that $T_j = 0$, therefore

$$\begin{aligned} 0 &= T_j \\ &= V_{j-1} - V_{j-1}(f(x,\mu_j)) - l(x,\mu_j(x)) \\ &= V_{j-1} - \min_u \{V_{j-1}(f(x,u)) + l(x,u)\} \end{aligned}$$

□

Moreover, if $\{\mu_j\}_{j\geq 1}$ is selected as in the last theorem we can establish a linear convergence rate

THEOREM 2.5—SPEED OF CONVERGENCE
Suppose that there is a parameter $\gamma > 0$ such that $V^*(f(x,u)) \leq \gamma l(x,u)$ for all (x,u) and that $V_0 \leq \delta V^*$. Define $\sup_j \alpha_j = \hat{\alpha}$. If $\{V_j, \mu_j, \alpha_j\}$ is selected according to Theorem 2.32 then for every j

$$V_j \leq (1 + (\delta - 1)\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j)V^*$$

□

PROOF 2.5
Fix $j \geq 1$ and assume that $V_{j-1} \leq \delta_{j-1} V^*$ for all x. First observe that for any numbers a and b

$$\delta_{j-1}a + b + (\gamma b - a)\frac{\delta_{j-1} - 1}{\gamma + 1} = \frac{\delta_{j-1}\gamma + 1}{\gamma + 1}(a + b)$$

Consider inequality (2.29)

$$\begin{aligned}
V_j(x) &\leq V_j(f(x,\mu_j(x))) + l(x,\mu_j(x)) \\
&\quad + \alpha_j(V_{j-1}(x) - V_{j-1}(f(x,\mu_j)) - l(x,\mu_j(x))) \\
&\leq V_j(f(x,\mu_j(x))) + l(x,\mu_j(x)) \\
&\quad + \hat{\alpha}(V_{j-1}(x) - V_{j-1}(f(x,\mu_j)) - l(x,\mu_j(x))) \\
&\leq \hat{\alpha}V_{j-1}(x) + (1-\hat{\alpha})(V_{j-1}(f(x,\mu_j)) + l(x,\mu_j(x))) \\
&= \hat{\alpha}V_{j-1}(x) + (1-\hat{\alpha})\min_u\{V_{j-1}(f(x,u)) + l(x,u)\} \\
&\leq \hat{\alpha}\delta_{j-1}V^*(x) \\
&\quad + (1-\hat{\alpha})\min_u\{\delta_{j-1}V^*(f(x,u)) + l(x,u) \\
&\quad + (\gamma l(x,u) - V^*(f(x,u)))\frac{\delta_{j-1} - 1}{\gamma + 1}\} \\
&= \hat{\alpha}\delta_{j-1}V^*(x) + (1-\hat{\alpha})\frac{\delta_{j-1}\gamma + 1}{\gamma + 1}V^*(x) \\
&= [\hat{\alpha}\delta_{j-1} + (1-\hat{\alpha})\frac{\delta_{j-1}\gamma + 1}{\gamma + 1}]V^*(x) \\
&= [\frac{\delta_{j-1}(\gamma + \hat{\alpha}) + 1 - \hat{\alpha}}{\gamma + 1}]V^*(x)
\end{aligned}$$

Hence

$$\delta_j = \frac{\delta_{j-1}(\gamma + \hat{\alpha}) + 1 - \hat{\alpha}}{\gamma + 1}$$

We can find δ_j by back substitution

$$\begin{aligned}
\delta_j &= \frac{\delta_{j-1}(\gamma + \hat{\alpha}) + 1 - \hat{\alpha}}{\gamma + 1} \\
&= \delta_{j-1}\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right] + \left[\frac{1 - \hat{\alpha}}{\gamma + 1}\right] \\
&= \delta_0\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j + \left[\frac{1 - \hat{\alpha}}{\gamma + 1}\right]\sum_{k=0}^{j-1}\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^k \\
&= \delta_0\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j + \left[\frac{1 - \hat{\alpha}}{\gamma + 1}\right]\frac{1 - \left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j}{1 - \left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]} \\
&= \delta_0\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j + 1 - \left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j
\end{aligned}$$

using $\delta_0 = \delta$, we get

$$\delta_j = 1 + (\delta - 1)\left[\frac{\gamma + \hat{\alpha}}{\gamma + 1}\right]^j$$

□

Observe that the case with $\hat{\alpha} = 0$ corresponds to exact policy-iteration, this case gives the fastest convergence according to the formula in the last theorem. Note also that at the other extreme, i.e. when $\hat{\alpha} = 1$, the formula gives no information about the convergence rate, in accordance with the assumption in Theorem 2.32.

2.5 Summary and Concluding Remarks

This chapter has presented a new approximate policy iteration algorithm. A key feature of the algorithm is the monotonic convergence, a highly desirable property of an approximation algorithm. Moreover, the fact that no differentiability assumptions on exact cost functions are made makes these results applicable to wide range of problems.

Using the main result, we deduced an algorithm for input affine polynomial systems with quadratic penalty on control variables. For this special case, the algorithm can use the sum of squares framework to verify

required inequalities. The examples given for this special case illustrates another important feature. The algorithm can be executed with a value function parametrization that can not represent the exact cost. Yet, we get an improved approximation in each iteration.

3

Value Iteration With Polynomial Parametrization

3.1 Introduction

In Chapter 1 Relaxed Value Iteration was defined by

$$\begin{aligned} \min_{u\in\mathbb{U}(x)}\{V_{N-1}(f(x,u)) + \beta l(x,u)\} &\leq V_N(x) \\ V_N(x) &\leq \min_{u\in\mathbb{U}(x)}\{V_{N-1}(f(x,u)) + \alpha l(x,u)\} \end{aligned} \tag{3.1}$$

It is obvious that to find a V_N that satisfies these two inequalities some structure must be imposed on f, l and $\mathbb{U}(x)$.

In this chapter we consider two types of optimal control problems, and we propose solutions with multivariate polynomials as value function parametrization. First we consider constrained control of discrete time systems with continuous control space. To be able to perform value iteration we impose certain convexity assumptions and propose a semi-grid-based technique to verify (3.1). The second type of problem we consider is control of discrete time systems with discrete control space, i.e. a switching system. We propose a simple state weighting relaxation technique.

3.2 Lower Bound Approximation

Our strategy for handling the lower bound

$$\min_{u\in\mathbb{U}(x)}\{V_{N-1}(f(x,u)) + \beta l(x,u)\} \leq V_N(x) \tag{3.2}$$

in Equation (3.1) is by piecewise approximations. Given a set of points $\{x_1, \ldots, x_s\} \subset \mathbb{X} \subset \mathbb{R}^n$. Let $\{\mathbb{X}_j\}$ be a partition of $\mathbb{X}$. Each member of such a partition is defined by the convex hull of m points,

$$\mathbb{X}_j = \text{conv}(v_{j1}, \ldots, v_{jm}), \text{ with } v_{jk} \in \{x_1, \ldots, x_s\}$$

here

$$n + 1 \leq m \leq s$$

Moreover, we construct these sets such that

$$\cup \mathbb{X}_j = \mathbb{X} \text{ and } \text{int}(\mathbb{X}_j \cap \mathbb{X}_i) = \emptyset, \quad j \neq i$$

and all points in $\{x_1, \ldots, x_s\}$ must be a vertex point of some $\mathbb{X}_j$. Having obtained such a partition, we define a piecewise affine approximation of a function $g : \mathbb{X} \to \mathbb{R}$ as

$$\hat{h}(x) = \sum_{i=1}^{m} g(v_{ji})\lambda(v_{ji}), \text{ for } x \in \mathbb{X}_j \tag{3.3}$$

with

$$x = \sum_{i=1}^{m} v_{ji}\lambda(v_{ji}), \quad \sum_{i=1}^{m} \lambda(v_{ji}) = 1, \quad \lambda(v_{ji}) \geq 0 \tag{3.4}$$

any $x \in \mathbb{X}$ can be written this way. We say that $\hat{h}$ is a piecewise affine approximation of g, PWAA for short, defined by the points $\{x_1, \ldots, x_s\}$. The functions that we will approximate are all convex, obtained through partial minimization as in the following

THEOREM 3.1
Suppose that the function $F : \mathbb{X} \times \mathbb{U} \to \mathbb{R}$ is jointly convex in the arguments and bounded below. Suppose also that $c : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^n$ is convex, define

$$\hat{\mathbb{U}}(x) = \{u : c(x, u) \leq 0, u \in \mathbb{U}\}$$

Then the function $g : \mathbb{X} \to \mathbb{R}$, defined by

$$g(x) = \inf_{u \in \hat{\mathbb{U}}(x)} F(x, u) \tag{3.5}$$

is convex. □

PROOF 3.1
Let $x_1, x_2 \in \mathbb{X}$ and $\epsilon > 0$, by definition there are two points $u_1 \in \mathbb{U}(x_1)$ and $u_2 \in \mathbb{U}(x_2)$ such that $F(x_1, u_1) \leq g(x_1) + \epsilon$ and $F(x_2, u_2) \leq g(x_2) + \epsilon$. For a given $x \in \mathbb{X}$ our assumption on c means that $u \in \hat{\mathbb{U}}(x)$ if and only if $c(x, u) \leq 0$. Then for any $0 \leq \lambda \leq 1$ we have

$$\lambda u_1 + (1 - \lambda) u_2 \in \mathbb{U}(\lambda x_1 + (1 - \lambda) x_2)$$

hence

$$\begin{aligned} g(\lambda x_1 + (1 - \lambda) x_2) \leq & F(\lambda x_1 + (1 - \lambda) x_2, \lambda u_1 + (1 - \lambda) u_2) \\ \leq & \lambda F(x_1, u_1) + (1 - \lambda) F(x_2, u_2) \\ \leq & \lambda (g(x_1) + \epsilon) + (1 - \lambda)(g(x_2) + \epsilon) \\ = & \lambda g(x_1) + (1 - \lambda) g(x_2) + \epsilon \end{aligned}$$

since $\epsilon > 0$ was arbitrary the result follows. □

The following corollary is an consequence of Theorem 3.1

COROLLARY 3.1
Consider value iteration, with $V_0 = 0$, for a problem with convex step cost l and linear dynamic update equation $f(x, u) = Ax + Bu$. If the state dependent constraint $\hat{\mathbb{U}}(x)$ can be represented as $\{u : c(x, u) \leq 0, u \in \mathbb{U}\}$ with c convex, then the optimal cost function V_N^* is convex for all $N \geq 0$. □

PROOF 3.2
Apply Theorem 3.1 with $F(x, u) = V_{N-1}(f(x, u)) + l(x, u)$ □

Since the exact computation of

$$g = \min_{u \in \mathbb{U}(x)} \{V_{N-1}(f(x, u)) + \beta l(x, u)\}$$

is not, in general, feasible, we need to find a computable approximation. Replacing g with an upper approximation h, inequality (3.2) becomes

$$g(x) \leq h(x) \leq V_N(x) \tag{3.6}$$

If this approximation is done in every step of iteration (1.16) the conclusion of Proposition 1.1 clearly holds. On the other hand, if $g(x) > h(x)$ at any point, this conclusion can not be drawn.

To construct such an upper approximation h, we should select an appropriate parametrization for h. To be useful, such a parametrization

should have certain properties. In particular, if a given approximation is not sufficiently tight it should be easy to improve it. Moreover, it would be desirable to known a priori that the approximation can be made arbitrarily good. We next show that the piecewise approximation defined above have all these properties

LEMMA 3.1
Suppose that $g : \mathbb{X} \to \mathbb{R}$ is convex. Let h be a PWAA of g, then

$$g(x) \leq h(x), \quad \forall x \in \mathbb{X} \tag{3.7}$$

□

PROOF 3.3
By definition of convex function. □

If we add any point to a given vertex set the new approximation will be better everywhere

LEMMA 3.2
Let g be convex. Suppose that h and $\hat{h}$ are PWAAs of g defined by $\mathbb{W}$ and $\hat{\mathbb{W}}$ respectively. If $\mathbb{W} \subset \hat{\mathbb{W}}$ then

$$\hat{h}(x) \leq h(x), \quad \forall x \in \mathbb{X} \tag{3.8}$$

□

PROOF 3.4
Since $\mathbb{X}$ is bounded $\text{epi}(h)$ and $\text{epi}(\hat{h})$ have the same extreme directions. If $x \in \text{epi}(h)$ then x is a conic combination of extreme directions plus a convex combination of its extreme point, i.e. the points in $\mathbb{W}$. The same is true for points in $\text{epi}(\hat{h})$ with convex combination taken from $\hat{\mathbb{W}}$. Since $\mathbb{W} \subset \hat{\mathbb{W}}$, it follows that $\text{epi}(h) \subset \text{epi}(\hat{h})$. Hence $\hat{h} \leq h$. □

Next we conclude that as the partition gets finer the approximations tend pointwise to the function to be approximated

THEOREM 3.2
Suppose that $g : \mathbb{X} \to \mathbb{R}$ is continuous. Let $\{h_k\}$ be a sequence of PWAA's of g with corresponding partitions $\{\mathbb{X}_{jk}\}$. If

$$\sup_j \text{diam}(\mathbb{X}_{jk}) \longrightarrow 0 \tag{3.9}$$

then

$$h_k(x) \longrightarrow g(x), \quad x \in \mathbb{X} \tag{3.10}$$

□

PROOF 3.5
Let $x \in \mathbb{X}$. In the sequence of partitions there is, by assumption, a subsequence $\{\mathbb{X}_{j_p k_p}\}$ of sets such that $x \in \mathbb{X}_{j_p k_p}$ and $\mathbb{X}_{j_p k_p} \supset \mathbb{X}_{j_{p+1} k_{p+1}}$ for all p. Define $\mathbb{S} = \cap_p \mathbb{X}_{j_p k_p}$. By compactness, $\mathbb{S}$ is a non-empty compact set. By construction $x \in \mathbb{S}$, suppose that there is another point $y \in \mathbb{S}$. Then, $\text{diam}(\mathbb{S}) > 0$ and $\mathbb{X}_{j_p k_p} \supset \mathbb{S}$ implies that $\text{diam}(\mathbb{X}_{j_p k_p}) \geq \text{diam}(\mathbb{S}) > 0$, which is a contradiction. Continuity shows that $h_k(x) \longrightarrow g(x)$. □

And finally, with an additional assumption the convergence is uniform

THEOREM 3.3
Let $g : \mathbb{X} \to \mathbb{R}$ be convex and continuous. Suppose that $\{h_k\}$ and $\{\mathbb{X}_{jk}\}$ are as in Lemma 3.2 with corresponding sets of definition $\{\mathbb{W}_k\}$, such that $\mathbb{W}_k \subset \mathbb{W}_{k+1}$. If $\mathbb{X}$ is compact, then

$$h_k \longrightarrow g$$

the convergence is in the supremum norm. □

PROOF 3.6
For any $\epsilon > 0$, define $\mathbb{S}_k = \{x : h_k(x) \geq g(x) + \epsilon\} \subset \mathbb{X}$. Being a subset of $\mathbb{X}$, $\mathbb{S}_k$ is bounded and since h_k and g are continuous it is also closed, i.e. compact. Now, as $\mathbb{W}_k \subset \mathbb{W}_{k+1}$ Lemma 3.2 implies $h_{k+1}(x) \leq h_k(x)$, hence $\mathbb{S}_{k+1} \subset \mathbb{S}_k$. For any $x \in \mathbb{X}$, consider the sequence of real numbers $\{h_k(x)\}$, by Theorem 3.2 it converges to $g(x)$. Therefore, there is an N such that for $k \geq N$ it holds $x \notin \mathbb{S}_k$, we conclude $x \notin \cap \mathbb{S}_k$ and this holds for all $x \in \mathbb{X}$. Since $\mathbb{S}_k$ are compact this can only hold if $\mathbb{S}_m = \varnothing$ for some m, and thus for all $k \geq m$. This shows that $0 \leq h_k(x) - g(x) < \epsilon$ for all $x \in \mathbb{X}$ and all $k \geq m$. This completes the proof. □

Convex Polynomials

Let $\mathbb{X} \subset \mathbb{R}^n$ be a convex set. Recall that a twice differentiable function $p : \mathbb{X} \to \mathbb{R}$ is convex if and only if

$$y^T \nabla^2 p(x) y \geq 0, \quad \forall (x, y) \in \mathbb{X} \times \mathbb{R}^n \tag{3.11}$$

It will be useful for us to restate this condition in the following equivalent form. Let $\mathbb{Y} \subset \mathbb{R}^n$ be any compact convex set containing 0 in its interior, then condition (3.11) is equivalent to

$$y^T \nabla^2 p(x) y \geq 0, \quad \forall (x, y) \in \mathbb{X} \times \mathbb{Y} \tag{3.12}$$

Now, if $p \in \mathbb{R}_d[x]$ then $h \in \mathbb{R}_d[x,y]$ defined by

$$h(x,y) = y^T \nabla^2 p(x) y \tag{3.13}$$

is a polynomial of the same degree as p in n more variables. If $\mathbb{X}$ and $\mathbb{Y}$ are defined appropriately, we can apply Theorem 1.3 or Theorem 1.4 to check if p is convex, using convex optimization.

The Full Algorithm

Although the algorithm is not restricted to systems with linear dynamics we will focus on that.

First, let us see how the constraints on control variables propagate. At time k the state propagates according to the dynamic equation

$$x(k+1) = Ax(k) + Bu(k), \quad k \geq 0 \tag{3.14}$$

Constraints on state and control variables are defined by polytopes

$$\mathbb{X} = \{x : \mathbf{1} - Cx \geq 0\}, \quad U = \{u : \mathbf{1} - Fu \geq 0\} \tag{3.15}$$

Both $\mathbb{X}$ and $\mathbb{U}$ are assumed to be non-empty, convex and compact with $(0,0) \in \text{int}(\mathbb{X} \times \mathbb{U})$. To fulfill the constraints we must ensure that $\mathbb{X}$ is invariant. If $x \in \mathbb{X}$ we must have $Ax + Bu \in \mathbb{X}$ after applying u, that is

$$\mathbf{1} \geq CAx + CBu$$

For $x \in \mathbb{X}$ the feasible control set $\mathbb{U}(x)$ is thus,

$$\mathbb{U}(x) = \{u : \mathbf{1} \geq CAx + CBu\} \cap \{u : \mathbf{1} - Fu \geq 0\}$$

Let $\{\mathbb{X}_j\}$ be a given partition of $\mathbb{X}$, with points of definition $\mathbb{W} = \{w_1 \ldots w_m\}$ and suppose that $V_{N-1} \in \mathbb{R}_d[x]$ satisfies (1.16). To find a PWAA that satisfies (3.6) we need to solve the following m problems

$$g(w_i) = \min_{u \in \mathbb{U}(w_i)} \{V_{N-1}(Aw_i + Bu) + \beta l(w_i, u)\}$$

where $i = 1..m$ and

$$U(w_i) = \{u : \mathbf{1} \geq CAw_i + CBu\} \cap \{u : \mathbf{1} - Fu \geq 0\}$$

We consider only the case with l a convex polynomial, thus these m problems are convex if $V_{N-1}(Ax+Bu)$ is convex, this is so if $V_{N-1}(x)$ is convex. Solutions to these problems can be obtained quickly if the degrees of the

involved polynomials are moderate. It remains to derive a convex feasibility problem using Theorem 1.3 or Theorem 1.4. Each polygon $\mathbb{X}_j$ can be written as

$$\mathbb{X}_j = \{x : x = w_{j1} + \mathbb{W}_j \lambda_j, \lambda_j \succeq 0, 1 \geq \mathbf{1}^T \lambda_j\}$$

here

$$\lambda_j = [\lambda_{j2}, \ldots, \lambda_{jm_j}]^T$$

and the columns in $\mathbb{W}_j = [w_{j2} \ldots w_{jm_j}]$ are the vertex points of $\mathbb{X}_j$. The obtained PWAA, after the minimization in (3.2), can then be written as

$$h_j(\lambda_j) = \sum_{k=1}^{m_j} \lambda_{jk} g(w_{jk}), \quad \lambda_{j1} = 1 \tag{3.16}$$

Now, to execute one iteration of (1.16) we need to find a $V_N \in \mathbb{R}_d[x]$ such that

$$\begin{aligned} h_j(x) &\leq V_N(x), \quad \forall x \in \mathbb{X}_j, \quad j = 1..p, \\ V_N(x) &\leq V_{N-1}(Ax + Bu) + \alpha l(x,u) \\ &\forall (x,u) \in \mathbb{X} \times \mathbb{U}(x) \\ 0 &\leq y^T \nabla_x^2 V_N(x) y, \quad \forall (x,y) \in X \times Y \end{aligned} \tag{3.17}$$

where p is the number of partitions. Observe that we can not have $V_{N-1}(0) = V_N(0) = h(0) = 0$ and $V_{N-1}(x) \leq h(x) \leq V_N(x)$ since h is not differentiable. To resolve this, a small set around the origin has to be removed from $\mathbb{X}$ when verifying

$$h_j(x) \leq V_N(x) \tag{3.18}$$

See the partition in Figure 3.1.

Let us take

$$\mathbb{Y} = \{y : Hy \leq \mathbf{1}\}$$

And parametrize V_N as,

$$\sum_{\beta_1 + \ldots + \beta_n \leq d} \nu_\beta x^\beta \tag{3.19}$$

Application of theorem 1.3, we find that the first inequality in (3.17) becomes

$$\begin{aligned} &\sum_{\beta_1 + \ldots + \beta_n \leq d} \nu_\beta [w_{j1} + \mathbb{W}_j \lambda_j]^\beta - h(\lambda_j) \\ =&\sigma_{j0} + \sigma_{j1} \cdot (1 - \mathbf{1}^T \lambda_j) + \sum_{k=2}^{m} \sigma_{jk} \cdot \lambda_{jk} \end{aligned}$$

with $\sigma_{jk} \in \Sigma[\lambda_j]$. In this equality expression both sides are a polynomial in variables λ_j. The variables defining the optimization problem are the

coefficients in the σ_{jk}'s and the coefficients $\{v_\beta\}$ in the polynomial V_N. The other two inequalities in (3.17) define similar equalities in the polynomial variables x, u and x, y respectively. We note in particular that this algorithm only requires a vertex representation of each polytope.

We illustrate the ideas with a small example.

EXAMPLE 3.1
Consider the problem of controlling the double integrator

$$x(k+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix} u(k), \quad k \geq 0$$

The set $\mathbb{X}$ was chosen as in Figure 3.1. The corresponding matrix in equation (3.15) is given by

$$C = \frac{1}{25} \begin{bmatrix} -1 & -5 \\ 1 & 5 \\ 1 & 0 \\ -1 & 0 \\ 0 & 5 \\ 0 & -5 \end{bmatrix}$$

In addition, the control signal was bounded as

$$-1 \leq u(k) \leq 1$$

It has been shown in [Gutman and Cwikel, 1986] that this system is controlled invariant on $\mathbb{X}$.

The step cost was chosen as

$$l(x,u) = |x|^2 + \frac{1}{100}|u|^2$$

During the iterations the degree of the V_N polynomials were fixed to 4. Moreover, the degree of the σ_{jk} polynomials were fixed to 2. The initial partition was chosen according to Figure 3.1.

The algorithm was iterated 25 times. During the iterations it was necessary to add points to the partition. At the last iteration the partition consisted of 40 pieces, making the computation intense. However, we were able to verify the inequality (1.18) on

$$\mathbb{X} \cap \{x : |x_1| \geq 4, \quad |x_2| \geq 2\}$$

with $\beta = 0.23$ and $\alpha = 7.1$. The resulting value function is shown in Figure 3.2. From any given initial state we can easily compute the open

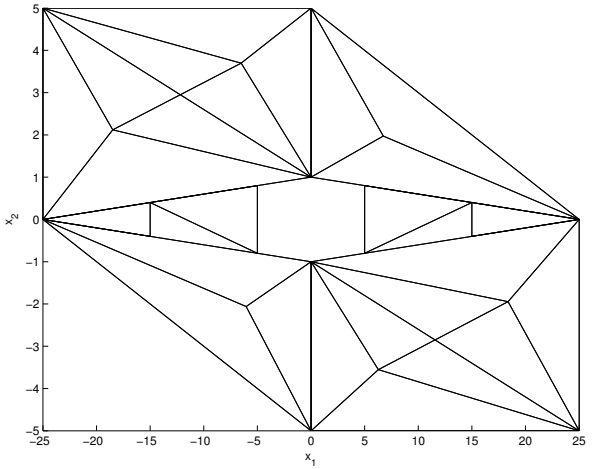


Figure 3.1 Initial partition of $\mathbb{X}$ in Example 3.1. The non-triangular region around the origin is removed when the inequalities (3.18) are verified.

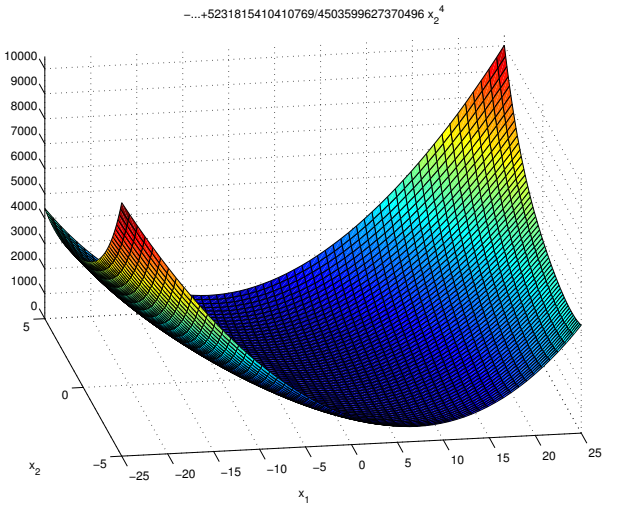


Figure 3.2 Approximate value function in Example 3.1.

loop optimal trajectory by solving a quadratic program. In Figure 3.3 we compare the open loop optimal trajectory with a trajectory obtained by using the just computed suboptimal feedback controller.

□

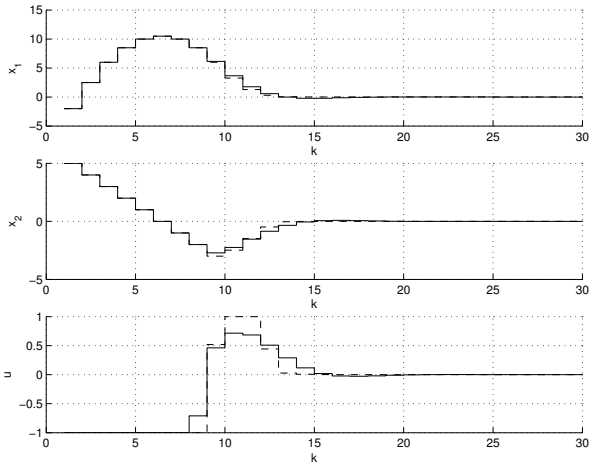


Figure 3.3 Dashed plots corresponds to the optimal open loop trajectory, and full plots to approximate feedback trajectory. The initial condition was $x(0) = [-2, 5]^T$. The state trajectories are almost overlapping everywhere but there is a slight difference in the control signal.

3.3 Discrete Control

In the rest of this chapter we assume that f and l are polynomials and that $\mathbb{X} = \{x : h_1(x) \geq 0, \ldots, h_p(x) \geq 0\}$ where each h_k is a polynomial. Let V_{N-1} be given and consider the upper inequality

$$V_N(x) \leq \min_{u \in \mathbb{U}(x)} \{V_{N-1}(f(x,u)) + \alpha l(x,u)\} \tag{3.20}$$

this inequality holds if and only if

$$V_N(x) \leq V_{N-1}(f(x,u)) + \alpha l(x,u), \quad \forall u \in \mathbb{U}(x) \tag{3.21}$$

Here we may consider the right hand side as polynomial in (x,u), and thus we may apply the results from Section 1.4 in Chapter 1 to obtain a finite dimensional constraint on V_N. However, as we are interested in switched problems in this section we now consider the case with a finite control set $\mathbb{U}$, say $|\mathbb{U}| = m$. Application of theorem 1.3 gives m finite dimensional constraints on V_N

$$-V_N(x) + V_{N-1}(f(x,u_k)) + \alpha l(x,u_k) = \sigma_{k0} + \sum_{j=1}^{p} h_j \sigma_{kj} \tag{3.22}$$

where each σ_{kj} is a sum of squares in x.

The lower inequality

$$g(x) = \inf_{u \in U(x)} \{V_{N-1}(f(x,u)) + l(x,u)\} \leq V_N(x) \tag{3.23}$$

is again more difficult, we need to approximate g from above. In the case of finite $\mathbb{U}$ we may consider a simpler approach, compared to the method proposed in the previous section. To this end, consider the set of all polynomial partitions of unity

$$\mathbb{W} = \{(w_1, .., w_m) : \sum_{k=1}^{m} w_k(x) = 1, \\ 0 \leq w_k(x) \ \forall x \in \mathbb{X}, \ w_k \in \mathbb{R}[x]\}$$

Then we have

PROPOSITION 3.1
Let $(w_1, .., w_m) \in \mathbb{W}$ then $\forall x \in \mathbb{X}$

$$g(x) \leq \sum_{k=1}^{m} w_k(x)[V_{N-1}(f(x,u_k)) + l(x,u_k)]$$

□

PROOF 3.7
For any $u_k \in \mathbb{U}$ we have by definition of g

$$g(x) \leq [V_{N-1}(f(x,u_k)) + l(x,u_k)], \quad x \in \mathbb{X}$$

multiplying both sides by $w_k(x) \geq 0$ and summing over k we get

$$g(x) = \sum_{k=1}^{m} w_k(x) g(x) \leq \sum_{k=1}^{m} w_k(x)[V_{N-1}(f(x,u_k)) + l(x,u_k)], \quad x \in \mathbb{X}$$

□

The idea is to relax the lower inequality (3.23) and replace it with

$$\sum_{k=1}^{m} w_k(x)[V_{N-1}(f(x,u_k)) + l(x,u_k)] \leq V_N(x), \ \forall x \in \mathbb{X}$$

And just as for the upper bound we can write this as

$$V_N(x) - \sum_{k=1}^{m} w_k(x)[V_{N-1}(f(x,u_k)) + l(x,u_k)] = \sigma_0 + \sum_{j=1}^{p} h_j \sigma_j$$

where each σ_j is sum of squares in x. This constraint on V_N together with equations (3.22) defines the constraints that the sequence $\{V_N\}$ in the relaxed value iteration must satisfy.

Note, that the strategy of replacing g with this weighted sum is a natural relaxation considering the following

PROPOSITION 3.2
Let $\mathbb{U}$ be compact and $F : \mathbb{U} \to \mathbb{R}$ continuous. Let $\mathbb{M}(\mathbb{U})$ be the set of all probability measures supported on $\mathbb{U}$. Then

$$\min_{u \in \mathbb{U}} F(u) = \min_{\mu \in \mathbb{M}(U)} \int F(u) d\mu(u)$$

□

Examples

The following two examples are both application of relaxed value iteration.

EXAMPLE 3.2
The following example is also used in [Lincoln, 2003], where the synthesis was done using relaxed dynamic programming with a very different parametrization of the value function. We shall see that the resulting control law is much simpler using the method proposed in this chapter. The problem involves a DC-DC converter, we will consider control synthesis for similar circuits in some detail in Chapter 6. Consider the continuous-time model

$$\dot{x}_1 = \frac{1}{c}(x_2 - i_{load})$$
$$\dot{x}_2 = \frac{1}{l}(-x_1 - r x_2 + s(t) v_{in})$$

where x_2 denotes current i through the inductor, x_1 denotes voltage v_c over the capacitor and $s(t) \in \{-1, 1\}$ is the sign of the switch. The primary control objective is to find a feedback switching sequence so that the load voltage is constant despite changes in load current and input load variations. To make it robust, integral action is added to the model

$$\dot{x}_3 = v_{ref} - x_1$$

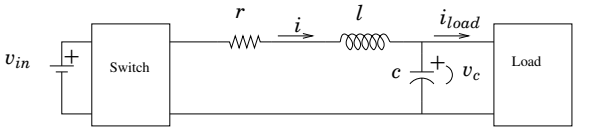


Figure 3.4 Circuit in Example 3.2.

Switching can only occur at a fixed sampling frequency, so the control problem is to select between two autonomous linear systems. After sampling, the system can be written as

$$x_e(k+1) = \Phi_i x_e(k)$$

with $x_e = [x^T \quad 1]^T$
A reasonable step cost is given by

$$l(x) = q_p(x_1 - v_{ref})^2 + q_i x_3^2 + q_d(x_2 - i_{load})^2$$

with positive weighting constants q_p, q_i and q_d. Note that $l > 0$ for all x except for $\hat{x} = [v_{ref}, i_{load}, 0]^T$ but $[\hat{x}, 1] \neq \Phi_1[\hat{x}, 1]^T \neq \Phi_2[\hat{x}, 1]^T$ so for any switching sequence the total cost becomes $\sum l(x(k)) = \infty$. Consider instead the average cost

$$V(x) = \sum_{k=0}^{\infty} \lambda^k l(x(k))$$

with $0 < \lambda < 1$. Actually, this is a (scaled) average since

$$1 = (1-\lambda)\sum_{k=0}^{\infty} \lambda^k$$

Sometimes λ is also called forgetting factor. Bellman's equation for the average cost function is

$$V^* = \min_u \{\lambda V^*(f(x,u)) + l(x,u)\}$$

Solving Bellman's equation with an average cost function using value iteration is similar to the case with total cost, we just multiply the appropriate term with λ.

We solve the problem for states in $\{x : 15 - |x|^2 \geq 0\}$. The constraints take the form

$$-V_N(x) + \lambda V_{N-1}(\Phi_1 x_e) + \alpha l(x) = \sigma_{10} + \sigma_{11}(15 - |x|^2)$$
$$-V_N(x) + \lambda V_{N-1}(\Phi_2 x_e) + \alpha l(x) = \sigma_{20} + \sigma_{21}(15 - |x|^2)$$

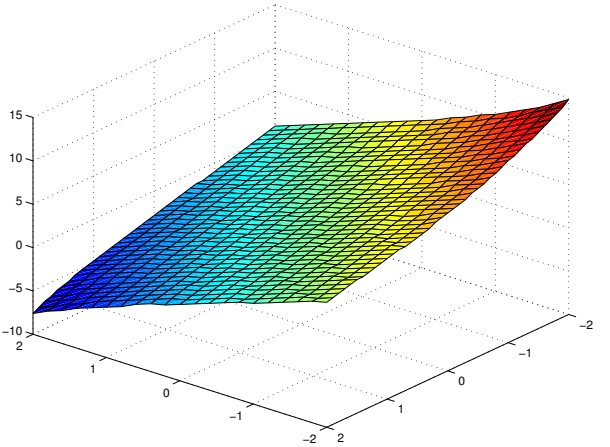


Figure 3.5 Each side of the plane corresponds to one switch position.

And the lower inequality becomes

$$V_N(x) - \sigma_{32}\lambda V_{N-1}(\Phi_1 x_e) - (1-\sigma_{32})\lambda V_{N-1}(\Phi_2 x_e) - l(x) \\ = \sigma_{30} + \sigma_{31}(15 - |x|^2)$$

with

$$1 - \sigma_{32} = \sigma_{40} + \sigma_{41}(15 - |x|^2)$$

all σ's being sum of squares in x.

After 50 iterations with $\lambda = 0.96$, $\alpha = 4.1$ and $\deg(V_k) = 4$ we have $V_N \approx V_{N-1}$. The controller which is given by

$$s(x) = \text{argmin}_{1,2}\{V_{50}(\Phi_1 x_e), V_{50}(\Phi_2 x_e)\}$$

is almost a switch-plane, see Figure 3.5. The performance of the closed loop is very similar to that in [Lincoln, 2003], but the controller appears much simpler. □

EXAMPLE 3.3
The following example is from [Koutsoukos and Antsaklis, 2002]. We consider the following switched discrete-time system

$$x(k+1) = A_q x(k), \quad q \in \{1,2\} \tag{3.24}$$

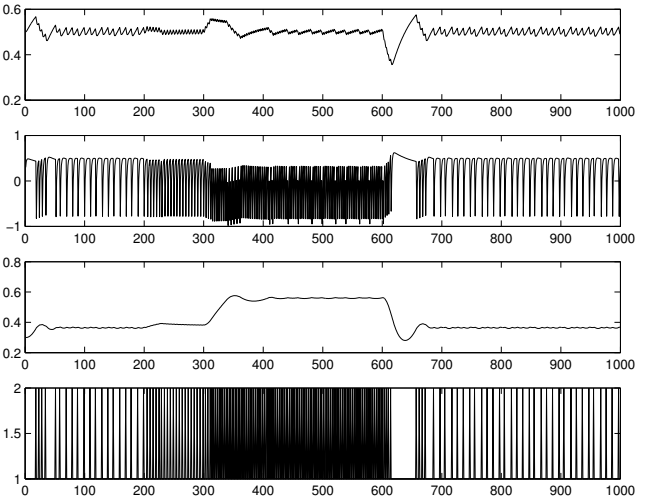


Figure 3.6 Top: output voltage. Next to top: current. Next to bottom: integral state. Bottom: Switch position. Reference voltage was $V_{ref} = .5$. At $k = 200$ the load current changes from its nominal value 0.3A to 0.1A, at $k = 300$ it changes to -0.2A and at $k = 600$ it changes back to its nominal value 0.3A.

where

$$A_1 = \begin{bmatrix} 1.7 & 4 \\ -0.8 & -1.5 \end{bmatrix}, A_2 = \begin{bmatrix} 0.95 & -1.5 \\ 0.75 & -0.55 \end{bmatrix}$$

We now consider the problem of computing a switching feedback controller for this system. We define the cost as

$$V(x) = \sum_{k=0}^{\infty} x(k)^T x(k)$$

Applying the proposed algorithm the equations are similar to those in the previous example. In this example $\mathbb{X} = \{x : 10 - |x|^2 \geq 0\}$. After 6 iterations V_N satisfy Proposition 1.2 with $\alpha = 1.7$ and $\deg(V_k) = 4$. The controller is given by

$$q(x) = \text{argmin}_{1,2}\{V_6(A_1 x), V_6(A_2 x)\}$$

The interesting thing about this example is that it is relatively simple to find a high performance feedback controller using the proposed relaxation. The closed loop trajectory that is shown in Figure 3.7 is almost identical to the one shown in [Koutsoukos and Antsaklis, 2002].

□

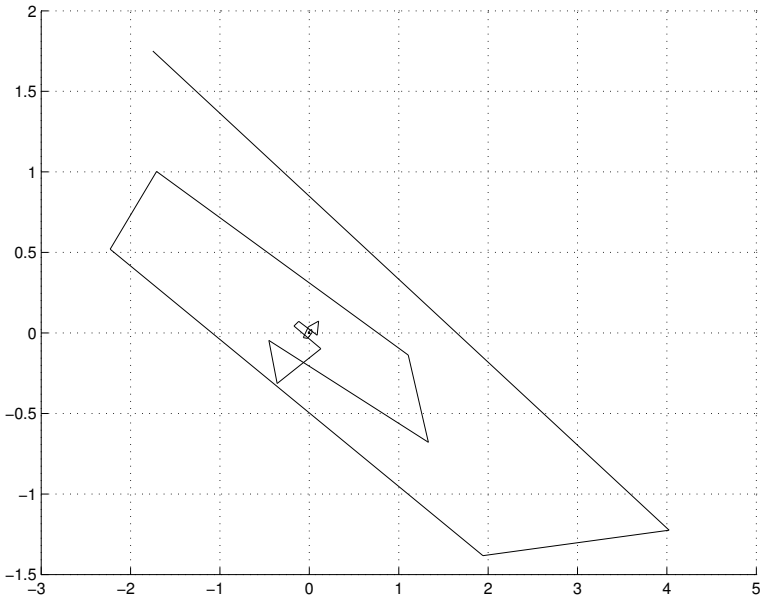


Figure 3.7 Closed loop trajectory starting from $x_0 = [-1.75\ 1.75]^T$. The trajectory cost is bounded form above by $1.7V^*(x_0)$.

3.4 Summary and Concluding Remarks

In this chapter, two new algorithms for approximate dynamic programming have been presented. The underlying approximation methodology is relaxed dynamic programming.

The first method applies to problems where each function in the lower bound sequence is convex. The main step is to approximate a convex function from above by piecewise linear functions and apply sum of squares techniques for verification of the resulting inequalities. The method can be thought of as a semi-grid-based technique. Due to the slack introduced by the relaxed dynamic programming formulation, the grid could be kept sparse. As a special case we considered a linearly constrained, linear system with quadratic step cost. Solving this problem exactly is intractable. With the proposed method, an approximately optimal solution, with bounds, could be computed. To make the method more efficient a systematic partitioning algorithm was needed.

The second problem considered was optimal control of switch systems. An algorithm was constructed for polynomial systems by combining a simple state weighting relaxation with the sum of squares framework for ver-

ification of inequalities. It was shown that this relaxation was natural from a theoretical point of view. Also, two examples were given to illustrate the method. In particular, the control law obtained in the DC-DC converter example appears to be simpler then previous result reported in the literature.

4

Constrained Control of Linear Systems

This chapter considers a Relaxed Dynamic Programming solution to the constrained optimal control problem of linear systems with convex piecewise linear step cost. The unconstrained problem has been considered before in [Lincoln, 2003; Lincoln and Rantzer, 2006], and therefore several parts of this chapter overlaps with that work. However, several extensions are presented. We incorporate constraints on states and control variables. A formula for an explicit state feedback controller is given. We also present a reduction technique that can be used to simplify the resulting controller lookup table. See also [Shamma and Xiong, 1997; Bemporad *et al.*, 2000], and the references therein, for related problems.

To compress notation we use

$$e(x) := \begin{bmatrix} x^T & 1 \end{bmatrix}^T$$
$$e(x,u) := \begin{bmatrix} x^T & u^T & 1 \end{bmatrix}^T$$

to denote extended vectors.

4.1 Optimal Control of Linear Systems

We start out by defining the class of step-cost functions considered in this section.

Convex Piecewise Linear Functions

Let $\mathbb{P}$ be a finite set of vectors in $\mathbb{R}^{n+1}$. The function

$$\max_{p\in\mathbb{P}} \langle p, e(x) \rangle \tag{4.1}$$

is convex. Any other convex function can be approximated to arbitrary precision using a max of linear function, a fact that makes this class of functions flexible when designing a step cost. Let $\mathbb{Q}$ be a finite set of vectors in $\mathbb{R}^{n+m+1}$. Consider the step-cost defined by

$$l(x,u) = \max_{q\in\mathbb{Q}} \langle q, e(x,u)\rangle \tag{4.2}$$

By choosing $\mathbb{Q}$ appropriately, l can be constructed to give high penalty on specific regions in state space. This can, for example, be used to approximate hard constraints with soft constraints. A special case of interest is when $\mathbb{Q}$ is chosen to represent a weighed l_1- or l_∞-norm. For example, a weighed l_∞-norm can be written as

$$\begin{aligned} ||We(x,u)||_\infty &= \max_j |w_j e(x,u)| \\ &= \max\{w_1 e(x,u), -w_1 e(x,u), ..., w_{n_W} e(x,u), -w_{n_W} e(x,u)\} \end{aligned}$$

where w_j denotes the rows in W. So in this case

$$\mathbb{Q} = \{w_1, -w_1, ..., w_{n_W}, -w_{n_W}\}$$

4.2 Problem Formulation

Consider a linear system

$$x(k+1) = \Phi x(k) + \Gamma u(k) + \nu, \quad k \geq 0 \tag{4.3}$$

where $\Phi \in \mathbb{R}^{n\times n}$, $\Gamma \in \mathbb{R}^{n\times m}$ and $\nu \in \mathbb{R}^n$. We consider problems were the state and control variable are required to satisfy linear constraints of the form

$$x \in \mathbb{X} = \{A_1 x \leq C_1\}, \quad u \in \mathbb{U} = \{B_1 u \leq C_2\} \tag{4.4}$$

Given any point $x \in \mathbb{X}$ the feasible control set is given by those $u \in \mathbb{U}$ such that

$$\Phi x + \Gamma u + \nu \in \{x : A_1 x \leq C_1\}$$

that is to say

$$\begin{aligned} u \in \mathbb{U}(x) &= \{u : B_1 u \leq C_2, \quad A_1(\Phi x + \Gamma u + \nu) \leq C_1\} &(4.5)\\ &:= \{u : Ax + Bu \leq C\} &(4.6) \end{aligned}$$

This set is assumed to be non-empty throughout this chapter, therefore system (4.3) is assumed to be controlled invariant on $\mathbb{X}$.

We are now ready to formally state the linearly constrained optimal control problem as

$$\min_{u[k]\in\mathbb{U}(x[k])} \sum_{k=0}^{\infty} \max_{q\in\mathbb{Q}} \langle q, e(x[k], u[k])\rangle, \quad \forall x[0] \in \mathbb{X}$$

subject to

$$x(k+1) = \Phi x(k) + \Gamma u(k) + \nu$$

4.3 Dynamic Programming Solution

We shall show that the following representation

$$V(x) = \max_{p\in\mathbb{P}} \langle p, e(x)\rangle, \quad x \in \mathbb{X} \tag{4.7}$$

is invariant under value iteration. To take one step of the value iteration algorithm we need to compute

$$V^+(x) = \min_{u\in\mathbb{U}(x)} \{V(\Phi x + \Gamma u + \nu) + l(x,u)\} \tag{4.8}$$

Where $\mathbb{U}(x)$ is given by (4.5). Let $\hat{\mathbb{P}}$ be the set of vectors given by

$$\hat{p} := \begin{bmatrix} \Phi & \Gamma & \nu \\ 0 & 0 & 1 \end{bmatrix}^T p, \quad \forall p \in \mathbb{P}. \tag{4.9}$$

Using this, we see that (4.8) becomes

$$V^+(x) = \min_{u\in\mathbb{U}(x)} \{\max_{p\in\hat{\mathbb{P}}} \langle p, e(x,u)\rangle + \max_{q\in\mathbb{Q}} \langle q, e(x,u)\rangle\} \tag{4.10}$$

Let $[Q_x \quad Q_u \quad Q_c]$ be the matrix obtained by stacking the elements of $\mathbb{Q}$ on top of each other, and similar for $\hat{\mathbb{P}}$. For fixed $x \in \mathbb{X}$ the value $V^+(x)$ is given by the optimal solution to the linear program

$$\min_{u,t_1,t_2} t_1 + t_2 \tag{4.11}$$

$$Q_x x + Q_u u + Q_c \leq t_1 \mathbf{1} \tag{4.12}$$

$$P_x x + P_u u + P_c \leq t_2 \mathbf{1} \tag{4.13}$$

$$Ax + Bu \leq C \tag{4.14}$$

with dual

$$\max_{(\lambda_q,\lambda_p,\lambda_f)\in E} \lambda_q^T(Q_x x + Q_c) + \lambda_p^T(P_x x + P_c) + \lambda_f^T(Ax - C) \tag{4.15}$$

$$\tag{4.16}$$

where E is the polytope

$$\begin{bmatrix} P_u^T & Q_u^T & B^T \\ \mathbf{1}^T & 0 & 0 \\ 0 & \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \lambda_p \\ \lambda_q \\ \lambda_f \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \tag{4.17}$$

$$\lambda_q, \lambda_p, \lambda_f \succeq 0 \tag{4.18}$$

Observe that E is independent of the state. Theorem 1.4 shows that the optimal value of the dual is attained at an extreme point of E. As a consequence, V^+ can be computed by forming the set $\mathbb{E}$ consisting of points

$$p = \begin{bmatrix} P_x & P_c \\ Q_x & Q_c \\ A & -C \end{bmatrix}^T \begin{bmatrix} \lambda_p \\ \lambda_q \\ \lambda_f \end{bmatrix} \tag{4.19}$$

for each extreme point $[\lambda_p^T \quad \lambda_q^T \quad \lambda_f^T]^T \in E$. Then by definition

$$V^+(x) = \max_{p\in\mathbb{E}} \langle p, e(x) \rangle$$

4.4 Computing the Control Law

The number of rows in the constraint matrix defining the polytope E is equal to $m + 2$, where m is the number of inputs, let us denote this matrix by H and the corresponding right hand side by h. In particular, this implies that every extreme point has at most $m + 2$ nonzero entries. Now let x be a given point in $\mathbb{X}$. If we evaluate V^+ at x the maximum is attained at a point $p \in \mathbb{E}$ with corresponding extreme point λ. Of course, we may assume that

$$\lambda = [\lambda_y^T \quad \lambda_z^T]^T = [\lambda_1 ... \lambda_{m+2} \quad 0..0]^T$$

with non-zero values only in the first $m+2$ entries. Let the corresponding partition of H be $H = [Y \quad Z]$, with $Y \in \mathbb{R}^{m+2\times m+2}$ invertible. Moreover,

let $c = [c_y^T \quad c_z^T]^T$ be the corresponding partition of the cost vector in (4.15), i.e. a partition of

$$\begin{bmatrix} P_x & P_c \\ Q_x & Q_c \\ A & -C \end{bmatrix} e(x) \tag{4.20}$$

LEMMA 4.1
If λ is optimal then

$$c_z^T \preceq c_y^T Y^{-1} Z$$

□

PROOF 4.1
By inserting

$$\lambda_y = Y^{-1}h - Y^{-1}Z\lambda_z$$

into the cost function we have

$$\begin{aligned} &c_y^T(Y^{-1}h - Y^{-1}Z\lambda_z) + c_z^T\lambda_z \\ &= c_y^T Y^{-1}h + (c_z^T - c_y^T Y^{-1}Z)\lambda_z \end{aligned}$$

Since $\lambda \succeq 0$ is optimal and we know that it is optimal to choose $\lambda_z = 0$, the result follows. □

THEOREM 4.1
If for a given $x \in \mathbb{X}$ the maximum in V^+ is attained at p, where p is given by (4.19), then the optimal feedback controller value is given by

$$\mu(x) = (-Y^{-T}c_y)_j, \quad 1 \le j \le m$$

where Y and c_y are defined as above. □

PROOF 4.2
First we show that the point $-Y^{-T}c_y$ is feasible for problem (4.11). $\begin{bmatrix} Y^T & Z^T \end{bmatrix}(-Y^T c_y) = \begin{bmatrix} -c_y & -Z^T Y^{-T} c_y \end{bmatrix}$, thus by the above lemma $[Y^T \quad Z^T](-Y^T c_y) \le [-c_y \quad -c_z]$, verifying the constraint in (4.11). To see that $-Y^{-T}c_y$ is optimal note that

$$-c_y^T Y^{-1}h = -c_y^T \lambda_y = -c^T \lambda \ge -h^T p \tag{4.21}$$

where, by weak duality, the last inequality holds for all p that are feasible for (4.11). The result follows by selecting $p^T = -Y^{-T}c_y$. □

Note that Y is a constant matrix and $c_y = We(x)$ where W is constant, therefore the corresponding controller can be written as

$$\mu(x) = L^T_{p(x)} e(x) \tag{4.22}$$

where $p(x) = \text{argmax}_{p \in \mathbb{P}} \langle p, e(x) \rangle$. It follows that the controller is piecewise linear.

Although the elements in the set $\mathbb{P}$ that defines V^+ are unique, it should be noted that the corresponding controller table need not be unique. The degree of non-uniqueness depends on the particular choice of step cost and constraint. A simple procedure to remove redundant entries in the controller is presented below in section 4.5.

Removing Redundancy in the Cost Table

Using the above technique to compute the value function by enumerating extreme points usually produces a lot of redundancy in the representation of V. There are usually points in $\mathbb{E}$ that do not attain the maximum in $\max_{p \in \mathbb{E}} \langle p, e(x) \rangle$ for any $x \in \mathbb{X}$. Such points are easy to find and discard. Let $w \in \mathbb{P}$ and $\hat{\mathbb{P}} = \mathbb{P} - \{w\}$ then

$$\max_{p \in \mathbb{P}} \langle p, e(x) \rangle = \max_{p \in \hat{\mathbb{P}}} \langle p, e(x) \rangle, \quad \forall x \in \mathbb{X} \tag{4.23}$$

if and only if there is no point $x \in \mathbb{X}$ such that

$$\langle w, e(x) \rangle > \langle p, e(x) \rangle, \quad \forall p \in \hat{\mathbb{P}} \tag{4.24}$$

That is true if and only if the linear program

$$\begin{aligned} L(\hat{\mathbb{P}}, w): \quad & \max_{t,x} \quad t \\ & \langle w, e(x) \rangle \geq \langle p, e(x) \rangle + t, \quad \forall p \in \hat{\mathbb{P}} \\ & A_1 x \leq C_1 \end{aligned}$$

has nonpositive optimal solution. Thus, a simple way to compute a minimal representation of a max of linear function is

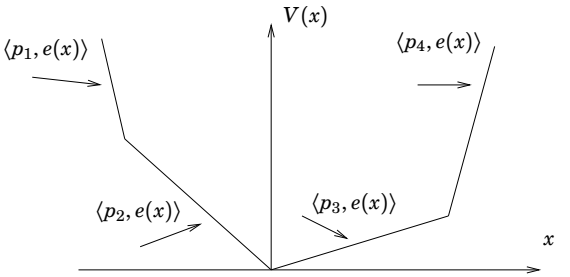


Figure 4.1 Redundancy illustration.

ALGORITHM 4.1

1: **for all** $w \in \mathbb{P}$ **do**
2: $\quad \hat{\mathbb{P}} \leftarrow \mathbb{P} - \{w\}$
3: $\quad q \leftarrow L(\hat{\mathbb{P}}, w)$
4: $\quad$ **if** $q \leq 0$ **then**
5: $\quad\quad \mathbb{P} \leftarrow \mathbb{P} - \{w\}$
6: $\quad$ **end if**
7: **end for**

□

A related question is if there exist an $x \in \mathbb{X}$ such that $V_1(x) > V_2(x)$, where $V_1 = \max_{p\in\mathbb{P}_1}\langle p, e(x)\rangle$ and $V_2 = \max_{p\in\mathbb{P}_2}\langle p, e(x)\rangle$ are two max of linear functions. This is a highly non-convex feasibility problem. Fortunately, the problem can easily be decomposed into a sequence of linear feasibility problems

ALGORITHM 4.2

1: **for all** $w \in \mathbb{P}_1$ **do**
2: $\quad$ solve $q \leftarrow L(\mathbb{P}_2, w)$
3: $\quad$ **if** $q > 0$ **then**
4: $\quad\quad$ return true
5: $\quad$ **end if**
6: **end for**

□

if label 4: is reached we have found an x such that $V_1(x) > V_2(x)$, otherwise $L(\mathbb{P}_2, w) \leq 0$ for all $w \in \mathbb{P}_1$ and consequently $V_1(x) \leq V_2(x)$ for all $x \in \mathbb{X}$.

4.5 Controller Reduction

As we have seen, the controller corresponding to a value function $V = \max_{p\in\mathbb{P}}\langle p, e(x)\rangle$ can be represented as

$$\mu(x) = L^T_{p(x)} e(x) \tag{4.25}$$

where $p(x) = \text{argmax}_{p\in\mathbb{P}}\langle p, e(x)\rangle$, the corresponding controller table is denoted by $\mathbb{C}$. We shall refer to an entry L_p in the controller table as redundant if L_p and the corresponding p can be removed from $\mathbb{C}$ and $\mathbb{P}$ respectively, in such a way that controller value $\mu(x)$ is equal before and after removal. As noted above, if we have computed $V = \max_{p\in\mathbb{P}}\langle p, e(x)\rangle$ the corresponding controller table $\mathbb{C}$ do not need not be unique. If the controller table is not unique one may suspect that some entries are redundant, as we shall see this may or may not be the case. The algorithm below can be motivated by two simple examples.

EXAMPLE 4.1
Consider the linear system

$$x(k+1) = x(k) + u(k)$$

where $x(k), u(k) \in \mathbb{R}$. If the step cost is

$$l(x,u) = || \begin{bmatrix} x \\ u \end{bmatrix} ||_\infty = \max\{|x|, |u|\}$$

the unconstrained optimal cost is $V^*(x) = |x|$ and the optimal controller is $\mu^*(x) = -x$ since

$$\begin{aligned} &\min_u\{|x+u| + \max\{|x|,|u|\}\} \\ =&\min\{\min_{\{u:|x|\geq|u|\}}\{|x+u|+|x|\}, \min_{\{u:|u|\geq|x|\}}\{|x+u|+|u|\}\} \\ =&|x| \end{aligned}$$

The corresponding tables are given by

$$\mathbb{P}^* = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}, \quad \mathbb{C}^* = \begin{bmatrix} -1 & 0 \\ -1 & 0 \end{bmatrix}.$$

Any one of the rows is redundant. □

EXAMPLE 4.2
Consider the situation in figure 4.1. When p_j is active, i.e. when x belongs to

$$\{x : \langle p_j, e(x)\rangle \geq \langle p_i, e(x)\rangle, \quad i \neq j\}$$

the corresponding controller value is $\langle L_{p_j}, e(x)\rangle$. Suppose that $L_{p_1} = L_{p_2} = L_{p_4}$ and that $L_{p_1} \neq L_{p_3}$, in particular L_{p_3} is not redundant. Moreover, if p_4 is removed from $\mathbb{P}$ all points x where p_4 would be active, will instead be active at p_3. Since, $L_3 \neq L_4$ it follows that L_{p_4} is not redundant. For the same reason, L_{p_2} is also not redundant. If p_1 is removed from $\mathbb{P}$ all points x where p_1 would be active will instead be active at p_2, since $L_{p_1} = L_{p_2}$, L_{p_1} is redundant. □

So essentially the idea is that it might be possible to reduce the complexity of the controller table by deleting neighboring control laws.

We now state this idea as a formal algorithm. Let $\cup\mathbb{C}_j = \mathbb{C}$ be a partition such that all elements in $\mathbb{C}_j$ are equal. Let $\cup\mathbb{P}_j = \mathbb{P}$ be the corresponding partition of the cost table. $A(p, \mathbb{P})$ denotes the active set of $p \in \mathbb{P}$, relative to $\mathbb{X}$, that is

$$A(p, \mathbb{P}) = \mathbb{X} \cap \{x : \langle p, e(x)\rangle \geq \langle \hat{p}, e(x)\rangle, \forall \hat{p} \in \mathbb{P} - \{p\}\}$$

As part of the algorithm below we will have to test if for a given s there is an $x \in A(p, \mathbb{P})$ such that

$$\langle s, e(x)\rangle > \langle p, e(x)\rangle, \quad \forall p \in \hat{\mathbb{P}} \tag{4.26}$$

That is true if and only if the linear program

$$\begin{aligned} L(A(p, \mathbb{P}), \hat{\mathbb{P}}, s): \quad & \max_{t,x} \quad t \\ & \langle s, e(x)\rangle \geq \langle p, e(x)\rangle + t, \quad \forall p \in \hat{\mathbb{P}} \\ & x \in A(p, \mathbb{P}) \end{aligned}$$

has positive solution.

We summarize the discussion in

ALGORITHM 4.3—REDUCTION

for each k **do**
  $\mathbb{S}_k \leftarrow \cup_{j \neq k} \mathbb{P}_j$
  for each $p \in \mathbb{P}_k$ **do**
    $\mathbb{P} \leftarrow \mathbb{P} - \{p\}$
    for each $s \in \mathbb{S}_k$ **do**
      $\hat{\mathbb{P}} \leftarrow \mathbb{P} - \{s\}$
      $q \leftarrow L(A(p, \mathbb{P}), \hat{\mathbb{P}}, s)$
      $\hat{\mathbb{P}} \leftarrow \mathbb{P} + \{s\}$
      if $q > 0$ **then**
        $\mathbb{P} \leftarrow \mathbb{P} + \{p\}$
        goto 3:
      end if
    end for
  end for
end for

□

4.6 The Complete Algorithm

Recall from Chapter 1 that at each iteration of relaxed value iteration we are given an approximate receding horizon cost function V. What we need to find is a $\hat{V}$ such that

$$V_l = \min_u \{V(f(x,u)) + \alpha l(x,u)\} \leq \hat{V}(x) \leq \min_u \{V(f(x,u)) + \beta l(x,u)\} = V_u$$

Now, if V is given by a max of linear functions we have seen in the previous section that V_l and V_u are also max of linear functions. Thus when these have been found we only have to find a V with the same representation satisfying $V_l \leq V \leq V_u$. Naturally, the goal is to find such a V with lower complexity than the lower and upper bounds. To find a V with the lowest possible complexity appears to be intractable from a computational point of view. The following non-optimal approach is simple, and works well as a compromise.

Given $V_l = \max_{p \in \mathbb{P}_l} \langle p, e(x) \rangle$ and $V_u = \max_{p \in \mathbb{P}_u} \langle p, e(x) \rangle$. Set $\mathbb{P} = \{0\}$.

ALGORITHM 4.4
1: **while** $\mathbb{P}_l \neq \emptyset$ and $\mathbb{P}_u \neq \emptyset$ **do**
2: Pick any $w \in \mathbb{P}_l$
3: $q \leftarrow L(\mathbb{P}, w)$
4: **if** $q \leq 0$ **then**
5: $\mathbb{P}_l \leftarrow \mathbb{P}_l - \{w\}$
6: **else**
7: $z \leftarrow \arg\max_{p \in \mathbb{P}_u} \langle p, e(x_w) \rangle$
8: $\mathbb{P} \leftarrow \mathbb{P} + \{z\}$
9: $\mathbb{P}_u \leftarrow \mathbb{P}_u - \{z\}$
10: **end if**
11: **end while**

□

The point x_w used on line 7: is the optimal solution found on line 5:. The algorithm simply adds elements from the upper bound until the lower bound is satisfied.

To prove that a function V is close to the stationary optimal cost function we need to verify

$$V_l = \min_u \{V(f(x,u)) + \alpha l(x,u)\} \leq V(x) \leq \min_u \{V(f(x,u)) + \beta l(x,u)\} = V_u$$

This can be done by simply applying Algorithm 4.2 two times, first to check $V_l \leq V$ and then $V \leq V_u$.

The full algorithm consists of using equation (4.8) to compute V_l and V_u and then applying Algorithm 4.4 to find an approximation. It is usually a good idea to reduce the approximation in each iteration, by applying Algorithm 4.3. When the close to stationary test above is successful the algorithm completes by computing and reducing the control law.

4.7 Examples

We give two examples to illustrate the ideas in this chapter. The algorithms have been implemented on top of the linear programming code CLP [CLP, 2007].

EXAMPLE 4.3
Consider a linear system with

$$\Phi = \begin{bmatrix} 1 & 1.1 \\ -1.1 & 1 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \nu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

We consider the unconstrained optimal control problem for this system with

$$l(x,u) = \| \begin{bmatrix} x \\ u \end{bmatrix} \|_\infty$$

The optimal stationary cost function and controller can be found after five value function iterations, they are given by

$$\mathbb{P}^* = \begin{bmatrix} 1.9690 & 3.4200 & 0 \\ -1.9690 & -3.4200 & 0 \\ 2.7590 & 2.6300 & 0 \\ -2.7590 & -2.6300 & 0 \\ 3.0091 & 2.2100 & 0 \\ -2.2000 & -0.2100 & 0 \\ -3.0091 & -2.2100 & 0 \\ 2.2000 & 0.2100 & 0 \\ -1.4196 & -3.7716 & 0 \\ 1.4196 & 3.7716 & 0 \end{bmatrix}, \quad \mathbb{C}^* = \begin{bmatrix} 0.0000 & -1.0000 & 0 \\ 0.0000 & -1.0000 & 0 \\ -1.0000 & 0.0000 & 0 \\ -1.0000 & 0.0000 & 0 \\ 0.1909 & -2.0000 & 0 \\ 0.1909 & -2.0000 & 0 \\ 0.1909 & -2.0000 & 0 \\ 0.1909 & -2.0000 & 0 \\ 0.6955 & -1.4450 & 0 \\ 0.6955 & -1.4450 & 0 \end{bmatrix},$$

Figure 4.2 shows the regions where the different entries in the optimal controller are active. μ_5 and μ_8 are equal and they are neighbors but Algorithm (4.3) fails to simplify the controller since according to our definition they are not redundant.

Now let us solve the same problem using relaxed value iteration. First we take $\alpha = 1.0$ and $\beta = 0.6$. The relaxed value iteration algorithm converges to a stationary solution in nine iteration. The approximation turns out to be very simple

$$\mathbb{P}^9 = \begin{bmatrix} -1.5582 & -0.3880 & 0 \\ -1.0687 & -2.5016 & 0 \\ -1.9135 & -1.4448 & 0 \\ 1.0687 & 2.5016 & 0 \\ 1.5582 & 0.3880 & 0 \\ 1.9135 & 1.4448 & 0 \end{bmatrix}, \quad \mathbb{C}^9 = \begin{bmatrix} 0.4079 & -1.7613 & 0 \\ 0.4079 & -1.7613 & 0 \\ 0.4079 & -1.7613 & 0 \\ 0.4079 & -1.7613 & 0 \\ 0.4079 & -1.7613 & 0 \\ 0.4079 & -1.7613 & 0 \end{bmatrix},$$

By inspection, an equivalent controller is given by

$$\mu^9(x) = \begin{bmatrix} 0.4079 & -1.7613 \end{bmatrix} x, \quad \forall x$$

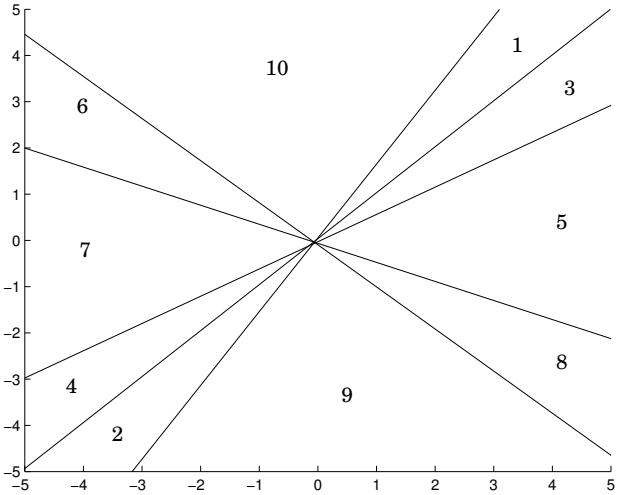


Figure 4.2 The numbers mark the regions where the corresponding entry in the optimal control table is active 4.3.

Note that the reduction algorithm can be used to deduce this single feedback gain from table 4.3. By using this linear feedback controller the closed loop system is guaranteed to have a performance given by

$$V^*(x) \leq V_{\mu_9}(x) \leq \frac{1}{0.6} V^*(x), \quad \forall x$$

This is, indeed, an interesting example showing the complexity versus performance trade-off. □

EXAMPLE 4.4
Consider a sampled double integrator

$$\Phi = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}, \quad \nu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

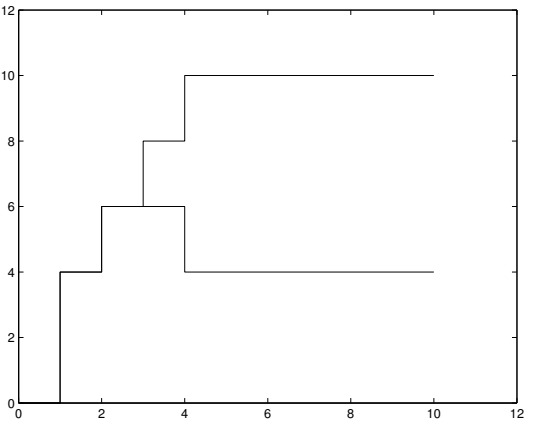


Figure 4.3 Cost function complexity during value function iteration. The upper curve corresponds to exact value iteration, the lower curve corresponds to relaxed iteration.

With constraints

$$\frac{1}{25}\begin{bmatrix} -1 & -5 \\ 1 & 5 \\ 1 & 0 \\ -1 & 0 \\ 0 & 5 \\ 0 & -5 \end{bmatrix} x \leq \mathbf{1} \text{ and } |u| \leq 1$$

Take the l_1-norm as step cost

$$l(x,u) = |x_1| + |x_2| + |u|$$

First an attempt to solve the problem using exact value iteration was made. The complexity of the value function during fifteen iterations is shown in Figure 4.4. The complexity grows monotonically. At the fifteenth iteration, with a table size of 405, the number of potential extreme points is about 10^7.

The lower curve in Figure 4.4 corresponds to the complexity during relaxed value iteration, $\beta = 0.7$ and $\alpha = \frac{1}{\beta}$. After 30 iterations the value function is stationary with a table size of 87. By applying the controller reduction procedure the controller table was reduced by 41% to size of 51.

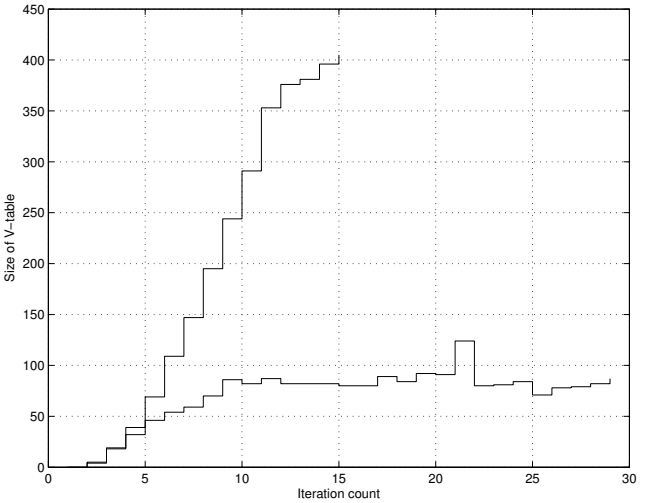


Figure 4.4 Cost function complexity during value function iteration. The upper curve corresponds to exact value iteration, the lower curve corresponds to relaxed iteration.

□

4.8 A Note on Complexity

For a polytope such as E in section 4.3 we have seen, in Chapter 1, that the maximum number of extreme points is $o(n^{m+2})$, where in the case of this chapter m is the number control variables. Thus, for a given synthesis problem it is only n that varies between each value iteration. The point of the relaxed dynamic programming algorithm is to introduce a slack so that the complexity of the value functions can be kept low, in other words to keep n small. We have seen in the examples that this is precisely what happens if we chose a sufficiently large slack. Thus, it is feasible to do extreme point enumeration in this case.

4.9 Summary and Concluding Remarks

This chapter has presented some extensions to the relaxed dynamic programming algorithms for linearly constrained linear systems with piecewise linear convex step cost. A formula for computing an explicit feedback

control law was given. A simple algorithm for reducing the complexity of the controller lookup table was also presented. Some interesting examples, illustrating the complexity and performance tradeoff, were given.

Experience from numerical computations shows that the algorithms in this chapter, in particular enumeration of extreme points, can be numerically unstable. Future work should address such problems.

The representation of the control law as presented in this chapter is not unique. For example, it is possible to represent the lookup table as a search for a particular region in statespace. It would be interesting to compare different representations from a complexity point of view. Also, different representations may have different robustness properties when it comes to lookup errors.

Extending the work in this chapter to include optimal control of linearly constrained *piecewise linear systems* would be very interesting. That class of problems appears to be much more difficult to solve than the ones considered in this chapter.

5

Dynamic Model Predictive Control

Model predictive control (MPC), see, e.g. [Maciejowski, 2002; Garcia *et al.*, 1989], has been used by the industry for several years, e.g. in the chemical industry and other industries where the processes have slow dynamics. The main reason for the success of MPC is the ability to control constrained multivariable systems. The MPC controller often relies on an online solution of a finite horizon optimal control problem, in *each* sample. Usually, the optimal control problem is reformulated as finite dimensional convex optimization problem. For example, a linear system with linear constraints on states and control variables and a quadratic step cost can be formulated as a quadratic optimization problem. There is a rich class of convex optimization problems which are guaranteed to be solvable in polynomial time. Even with increasing computer power, on systems with fast dynamics this is not fast enough for MPC. Since, if the system must be sampled too fast, the MPC controller will not be able to finish its required computations in time. This is one reason why model predictive control is most successful on slow processes.

Moreover, in practice the optimization solver can have quite unpredictable execution times, the computations can exceed the time of one sample or even a few samples. This obviously leads to reduced performance of the system and might even lead to instability. One approach to handle such problems has been studied in [Heriksson, 2006], see also the references therein.

Work that relate to ours can be found in [Löfberg, 2003] where the author studies robust MPC. This approach was later extended in [Goulart *et al.*, 2006]. These two references consider problems that are similar to the ones considered in this chapter, but our approach is different from theirs.

To compensate for computational delays, a new approach is presented

in this chapter. Instead of calculating an optimal sequence of control inputs, an optimal dynamic controller is computed in each sampling instance. With the use of the Youla-parametrization, [Youla *et al.*, 1976a; Youla *et al.*, 1976b], the MPC problem in its original form can be reformulated to depend on the parameters of the controller in such a way that the optimization problem can be solved with the same complexity as the original problem. Without computational delays, the resulting dynamic controller can be shown to be equivalent with the controller obtained from the original MPC formulation. With this setup, when computational time delays occur, there is now a feedback controller which controls the process. This will improve performance since the system operates in closed loop at all times. This is contrary to traditional MPC, which will operate in open loop.

The outline of this chapter is as follows: In section 5.1 a short introduction to traditional MPC is given. The main idea of the chapter is presented in section 5.2. The structure of the controller is described and the MPC formulation is revised to fit the new structure. Difficulties that arise when computational delays are present are discussed. A case study of a double integrator is studied in section 5.3. In section 5.5 a short summary of the required on-line computation needed in Dynamic MPC is given.

5.1 Traditional MPC

Consider the discrete, time-invariant, linear plant P

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma_1 w(k) + \Gamma_2 u(k) \\ z(k) &= C_1 x(k) + D_{11} w(k) + D_{12} u(k) \\ y(k) &= C_2 x(k) + D_{21} w(k) \end{aligned}$$

where u is the control input and w the disturbances. The z vector is referred to as the controlled output, it contains those signals that we will include in the system performance index and those that we will put constraints on. For example states, tracking errors or control variables. The output y is the measured output that can be used for feedback. Notice that we do not allow a direct term of the control input to y. This is due to the fact that to determine $u(k)$, a measurement of $y(k)$ is needed and hence $y(k)$ cannot depend on $u(k)$.

In model predictive control a finite horizon optimal control problem is solved in each sample. The cost function is defined as

$$V(x(0), \mathbf{u}) = \sum_{i=0}^{N-1} \ell(z(i)) + F(x(N)) \tag{5.1}$$

where $z(i)$ are the controlled outputs of P when the control sequence $\mathbf{u} = (u(0), \ldots, u(N-1))$ is applied to the system with initial state $x(0)$. The functions ℓ and F are the stage cost and the terminal cost, respectively. The stage cost ℓ is assumed to satisfy $\ell(z) \geq c|z|^2$. Notice that if a terminal cost is included in the cost, it operates on the terminal state.

The objective in MPC is to minimize the cost function (5.1) with respect to the control sequence $\mathbf{u}$, subject to constraints on both state variables and the control sequence

$$\begin{aligned} &\min_{\mathbf{u}} V(x(0), \mathbf{u}) \\ &u(i) \in \mathbb{U}, 0 \leq i < N \\ &x(i) \in \mathbb{X}, 0 \leq i \leq N \end{aligned} \tag{5.2}$$

where $\mathbb{U}$ and $\mathbb{X}$ are the sets of allowed control sequences and states, respectively. To be able to guarantee a unique optimum, $\mathbb{U}$ is usually a convex, compact set and $\mathbb{X}$ a convex, closed set, each with the origin included. When the optimal control sequence $\mathbf{u}^0$ has been determined, only the first control input $u^0(0)$ is applied to the plant P and the MPC procedure is repeated in the next sample.

If the step cost is convex and piecewise linear or quadratic and the sets $\mathbb{U}$ and $\mathbb{X}$ are convex polytopes the optimization problem (5.2) is convex and can be solved in a relatively efficient way.

5.2 Dynamic MPC

The idea of Dynamic MPC, which is presented below, is similar to traditional MPC. The main goal is to obtain an optimal control sequence which minimizes a certain cost function. The difference is that instead of directly calculating an optimal control sequence in each sample, an optimal dynamic controller is computed. Moreover, we do not optimize over the control values directly, instead these are parametrized via a dynamic compensator which in turn is linearly parametrized in a finite number of parameters. Our optimization goal is the same as before, i.e. to solve (5.2) in each step. We will give the details below.

Formulation and Setup

If the plant P is both stabilizable and detectable, there exist matrices K and L such that both $\Phi - \Gamma_2 L$ and $\Phi - KC_2$ are stable. It is known, see [Youla *et al.*, 1976a; Youla *et al.*, 1976b; Boyd and Barratt, 1991], that

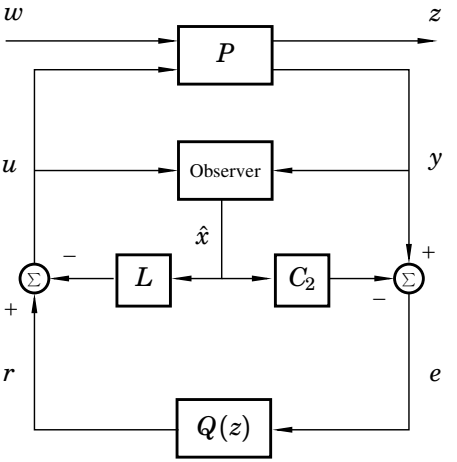


Figure 5.1 Block diagram of the Youla-parametrization.

the observer based nominal controller

$$\begin{aligned} \hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma_2 u(k) + Ke(k) \\ u(k) &= r(k) - L\hat{x}(k) \\ e(k) &= y(k) - C_2\hat{x}(k) \end{aligned} \tag{5.3}$$

combined with $r = Q(z)e$ for a stable $Q(z)$, gives a stable system. If Q is viewed as parameter, this construction is called the Youla-parametrization or Q-parametrization. A diagram of the system can be found in Figure 5.1.

An important condition in the Youla-parametrization is that the transfer function $T_{er} \equiv 0$, i.e. the transfer function from r to e is equal to zero. This condition is easily verified for the system described by P with the controller in (5.3). Using this condition, the transfer function of the system can be expressed as

$$G_{zw}(z) = T_{zw}(z) + T_{zr}(z)Q(z)T_{ew}(z)$$

where $T_{zw}(z)$, $T_{zr}(z)$ and $T_{ew}(z)$ are the transfer functions of the system when $Q(z)$ is removed. An illustrative diagram is found in Figure 5.2.

Since the MPC procedure is performed in each step, and since the initial state is changed in every step, care has to be taken when the initial

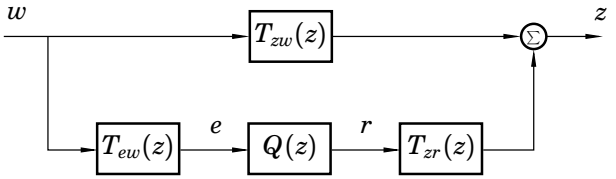


Figure 5.2 Block diagram of the Youla-parametrization in the general form.

state of the system does not equal zero. Consider the system

$$\begin{aligned} x(k+1) &= Ax(k) + B_1 w(k) + B_2 r(k) \\ z(k) &= C_1 x(k) \\ e(k) &= C_2 x(k) \end{aligned}$$

If the system has the initial state $x(0) \neq 0$ then, at time k, the outputs are

$$\begin{aligned} z(k) &= C_1\{A^k x(0) + \mathcal{B}_1 \mathbf{w}\} + C_1 \mathcal{B}_2 \mathbf{r} \\ e(k) &= C_2\{A^k x(0) + \mathcal{B}_1 \mathbf{w}\} + C_2 \mathcal{B}_2 \mathbf{r} \end{aligned} \tag{5.4}$$

where $\mathcal{B}_1$ and $\mathcal{B}_2$ are appropriate matrices, $\mathbf{w} = [w(0), \ldots, w(k-1)]^T$ and $\mathbf{r} = [r(0), \ldots, r(k-1)]^T$. It follows from (5.4) that the initial state $x(0)$ should only be associated with one of the input transfer functions. Since $T_{er} \equiv 0$, the initial state must be associated with the transfer function for w, i.e. the transfer functions $T_{zw}(z)$ and $T_{ew}(z)$.

Consider the plant P with the controller (5.3) combined with a $Q(z)$ given by

$$Q(z) = q_0 + q_1 z^{-1} + \ldots + q_{N-1} z^{-(N-1)}$$

where $q_j \in \mathbb{R}^{n_u \times n_y}$. It is clear that the closed loop is affine in the parameters q_j. In fact we shall show in section 5.5 that given an initial condition $x(0)$, the output at time i can be written as

$$z(i) = t(i) + h(i)\mathbf{q} \tag{5.5}$$

where $\mathbf{q}$ contains all the parameters in the filter $Q(z)$. Consider now the following optimization problem

$$\begin{aligned} &\min_{\mathbf{q}} V(x(0), \mathbf{q}) \\ &u(i) \in \mathbb{U}, 0 \leq i < N \\ &x(i) \in \mathbb{X}, 0 \leq i \leq N \end{aligned} \tag{5.6}$$

where the cost function is given by

$$V(x(0), \mathbf{q}) = \sum_{i=0}^{N-1} \ell(z(i)) + F(x(N)) \tag{5.7}$$

and each $z(i)$ is given by (5.5).

From an optimization point of view the two problems (5.6) and (5.2) are similar, in particular in the standard convex MPC formulation they are equally easy to solve. Moreover, these two problems are equivalent from a control point of view

THEOREM 5.1
Let $V_d^*(x(0))$ and $u_d^*(i)$ be the optimal cost and input trajectory corresponding to problem (5.6), let $V_s^*(x(0))$ and $u_s^*(i)$ be the optimal solution to (5.2). Then

$$V_d^*(x(0)) = V_s^*(x(0)), \text{ and } u_d^*(i) = u_s^*(i), \quad 0 \leq i \leq N-1$$

□

PROOF 5.1
Let $x_s^*(k)$, $0 \leq k \leq N$ be the states of P corresponding to the input sequence $\mathbf{u}^* = (u_s^*(0), \ldots, u_s^*(N-1))$. What has to be shown is that there exists a

$$Q^*(z) = q_0^* + q_1^* z^{-1} + \ldots + q_{N-1}^* z^{-(N-1)}$$

such that the system with the controller described by (5.3) also produces the input sequence $\mathbf{u}^*$, since if the sequence is the optimum of (5.2) then $Q^*(z)$ must be the optimum of (5.6).

Let $\tilde{x}^*(k) = x^*(k) - \hat{x}^*(k)$, it can be shown that

$$e^*(k) = C_2 \tilde{x}^*(k) + D_{21} w(k)$$

Define $e_k^* = (e^*(0), \ldots, e^*(k))$ for $0 \leq k < N$. If we use $u_d^*(k) = r^*(k) - L\hat{x}^*(k)$ and $r^* = Q^*(z)e^*$, it is easy to see that $Q^*(z)$ can chosen to satisfy

$$u_d^*(k) + L\hat{x}^*(k) = Q^*(z)e_k^*$$

□

REMARK 5.1
An implication of the theorem is that stability is insured under the same conditions as for the traditional MPC with the same cost function and constraints. Such conditions can be found in e.g. [Mayne *et al.*, 2000]. □

If constraints on the parameters of $Q(z)$ are included, the system will be bounded-input bounded-output stable.

THEOREM 5.2
Assume that for the plant P with initial state $x(0)$, the problem (5.6) combined with the constraints $|q_i| \leq c_q$, $0 \leq i \leq N-1$, is feasible for all times $k \geq 0$. The resulting system when controlling P with Dynamic MPC is BIBO stable. □

PROOF 5.2
Let $Q_k^0(z)$ be the optimal solution to (5.6) at time k. As will be seen in section 5.5

$$G_{zw}(z) = \left[\begin{array}{c|c} A & B \\ \hline C_k & D_k \end{array}\right]$$

where A and B are constant for all times and C_k and D_k are linearly dependent on the parameters of $Q_k^0(z)$. Since the Youla-parametrization gives stable system, it is obvious that for bounded inputs $w(k)$ the states $x(k)$ will be bounded. By the restriction of the parameters of $Q_k^0(z)$ it is also clear that $\|C_k\|_2 \leq c$ and $\|D_k\|_2 \leq d$ (for some c and d), for all $k \geq 0$. This gives

$$\begin{aligned} \|z(k)\|_2 &\leq \|C_k\|_2 \cdot \|x(k)\|_2 + \|D_k\|_2 \cdot \|w(k)\|_2 \\ &\leq c\|x(k)\|_2 + d\|w(k)\|_2 \end{aligned}$$

Hence the system is BIBO stable. □

Since the Dynamic MPC is equivalent to traditional MPC, when there are no computational delays, the advantages of Dynamic MPC shows up when such delays are introduced. Assume that at some time k, the time required to find the optimal solution to the MPC problem is longer than the sample time. Not to leave the system uncontrolled, the optimal solution from the previous time instant $k-1$ needs to be used. A straight-forward strategy in traditional MPC is to let the second control signal in the optimal control sequence be used as input to the system. This strategy is relying on open loop control, since the input does not depend on current measurement of the output. Dynamic MPC uses feedback to determine each control input. That is, even though no new optimal solution has been found yet, this strategy takes into account recent measured outputs when computing the next input. Hence, deviations of the measured outputs from the predicted outputs will be taken into account when the input is determined.

A FIR-filter $Q(z)$ that is not ready at the time interval it was initially supposed to be optimal for, is in some sense outdated. Since the filter is optimized for the initial state at the time instant when the optimization began, it is most likely not optimal at the current state. To improve the performance of the system, we can update the complete controller by simulating it for the time when the optimization took place. Assume that the

filter is delayed d samples and let $u_{\text{meas}}(k)$ be the inputs to the plant and $y_{\text{meas}}(k)$ be the measured outputs of the plant for the time during which the filter is being calculated, i.e. $0 \leq k < d$. Using the representation of the observer in (5.3), the update is performed according to

$$\begin{aligned} \hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma_2 u_{\text{meas}}(k) + Ke(k) \\ x_Q(k+1) &= A_Q x_Q(k) + B_Q e(k) \\ e(k) &= y_{\text{meas}}(k) - C_2\hat{x}(k) \end{aligned}$$

where A_Q and B_Q come from the state space representation of $Q(z)$. Now $\hat{x}(d)$ and $x_Q(d)$ are the updated states that should be used to initialize the controller.

5.3 Example

To illustrate the presented ideas, an example of the double integrator will be examined. The double integrator is

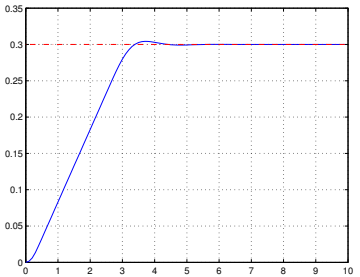
$$\begin{aligned} \dot{x} &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} x \end{aligned}$$

which is discretized with a sample interval of $h = 0.1$s. The discrete model is set up according to Figure 5.1, such that w contains the reference value, r, and z contains both y and the tracking error, $r - y$. The objective is to minimize the cost

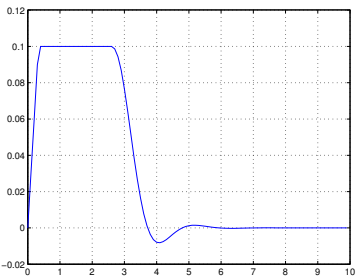
$$\sum_{k=1}^{N} (r(k) - y(k))^2 + \sum_{k=1}^{N} p \cdot (\Delta u(k))^2$$

under constraints on the velocity, $|x_2| \leq 0.1$, and on the inputs, $|u| \leq 0.3$. The position y is to follow the reference trajectory $r = 0.3$. $\Delta u(k)$ is the difference between the current and the previous input signal, i.e. $\Delta u(k) = u(k) - u(k-1)$. The prediction horizon N is set to 30 and the weight $p = 0.3$.

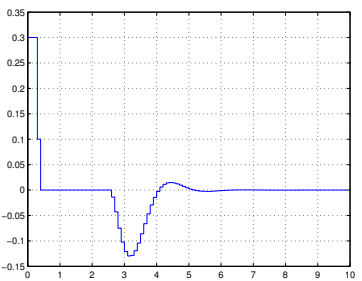
As seen by Proposition 5.1, if there is no computational delay, the system will be equivalent to a system controlled by a traditional MPC controller. The response of the reference trajectory $r = 0.3$ is found in Figure 5.3(a). The corresponding velocity and control input are found in Figures 5.3(b) and 5.3(c), respectively.



(a) The position of the double integrator.



(b) The velocity of the double integrator.



(c) The input to the double integrator.

Figure 5.3 Plots of the position, velocity and input of the double integrator controlled with traditional or Dynamic MPC, without computational delay.

Now, suppose that there is a constant computational delay of 5 samples, i.e. if the optimization starts at time k, it will not be finished until time $k + 5$. Note that a constant computational delay is often not encountered. Depending on which constraints are active, the computations

in the optimization take different time to perform. But varying computational delays will not in general be different compared to constant delays, assuming that the controller does not know that the computational delay is not constant.

If traditional MPC is used to control the process, with the strategy that for an input sequence $\mathbf{u}$ computed for time k (and therefore ready to be used at time $k+5$), the input that is used at time $k+i$ for $5 \leq i \leq 9$, is $u(i)$. The resulting system will be unstable. The response to the step $r(k) = 0.3$ is shown in Figure 5.4(a). The velocity and the applied control signal can be found in Figures 5.4(b) and 5.4(c).

Now, consider a Dynamic MPC controller. A controller that is computed during the time interval $[k\ k+5]$ is used to control the process during the interval $[k+5\ k+9]$. Before the controller can be used, at time $k+5$, the controller states must be updated. This is done as explained in the last section.

The position is shown in Figure 5.5(a), the reference was set to 0.3. Figure 5.5(b) shows the velocity of the system and Figure 5.5(c) shows the control signal.

5.4 Summary and Concluding Remarks

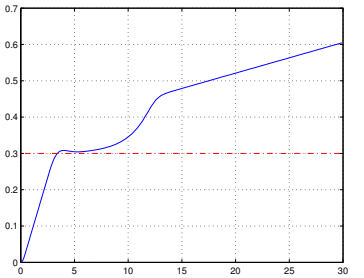
In this chapter, a dynamic version of MPC was presented. We showed that the presented method is equivalent to the standard formulation of MPC in the case when no computational delays are present. The idea is to update a feedback controller instead of an open loop trajectory, as in standard MPC.

We presented an illustrating example that showed the differences between two strategies to handle computational delays. The open loop solution became unstable whereas the Dynamic MPC solution achieved performance similar to the case with no delay.

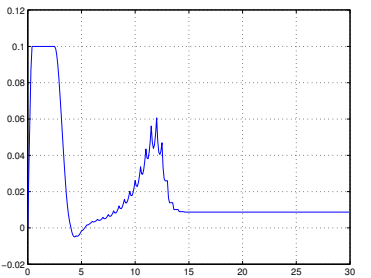
We believe that these preliminary results and ideas can be useful when approaching the problem with computational delays.

5.5 Appendix: The Online Optimization Problem

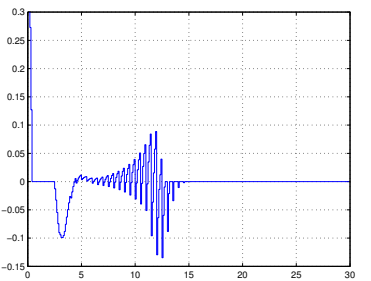
The purpose of this section is to give a brief review of how to exploit the Youla-parametrization for numerical computation, see [Boyd and Barratt, 1991; Boyd *et al.*, 1988]. In particular, we will show that the computational work required to solve the Dynamic MPC problem is similar to the work required in the traditional MPC formulation.



(a) The position of the double integrator.



(b) The velocity of the double integrator.



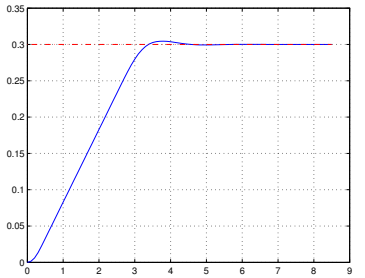
(c) The input to the double integrator.

Figure 5.4 Plots of the position, velocity and input of the double integrator controlled with traditional MPC, with a constant computational delay of 5.

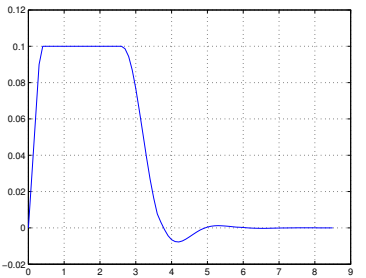
We have seen that the closed loop of any stabilizable linear system takes the form

$$G_{zw}(z) = T_{zw}(z) + T_{zr}(z)Q(z)T_{ew}(z)$$

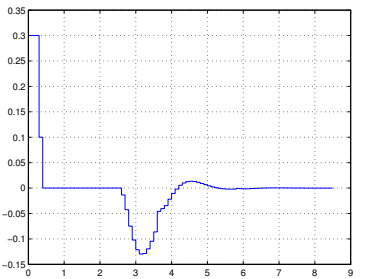
where $T_{zw}(z), T_{zr}(z)$ and $T_{ew}(z)$ are stable LTI-systems, depending only



(a) The position of the double integrator.



(b) The velocity of the double integrator.



(c) The input to the double integrator.

Figure 5.5 Plots of the position, velocity and input of the double integrator controlled with Dynamic MPC, with a constant computational delay of 5.

on the plant P and the nominal controller. Now consider the mapping

between w_j and z_i

$$\begin{aligned} G_{z_i w_j}(z) &= T_{z_i w_j}(z) + T_{z_i r}(z) Q(z) T_{e w_j}(z) \\ &= T_{z_i w_j}(z) + \sum_{k=1}^{n_u} \sum_{s=1}^{n_y} Q_{ks}(z) T_{z_i r_k}(z) T_{e_s w_j}(z) \end{aligned}$$

For the discussion in this section we may assume that $n_u = 1$ and $n_y = 1$, it is trivial to go from this case to the general MIMO case. Moreover, we will drop the channel indices for notational simplicity. Accordingly, we denote the transfer function for any channel by

$$G_{zw}(z) = T_{zw}(z) + Q(z) T_{zr}(z) T_{ew}(z)$$

with scalar transfer functions. The only constraint on the free parameter Q is that it should be a stable rational LTI transfer function, i.e. $Q \in \mathbb{RH}_\infty$. Thus the search space is infinite dimensional. To do numerical computations we need to restrict the search to a finite dimensional subspace. A simple choice is a FIR-base, z^{-l} for $0 \le l \le N-1$. Consider the candidate parametrization

$$Q(z) = \sum_{l=0}^{N-1} q_l z^{-l} = \left[\begin{array}{c|c} A_Q & B_Q \\ \hline C_Q & D_Q \end{array}\right]$$

where A_Q is the right shift matrix, B_Q is the first unit vector and

$$C_Q = [\, q_1 \quad q_2 \quad \dots \quad q_{N-1} \,], \quad D_Q = q_0$$

Note that A_Q and B_Q are fixed. Let

$$T_{zr}(z) T_{ew}(z) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

then we can take

$$\begin{aligned} H = Q(z) T_{zr}(z) T_{ew}(z) &= \left[\begin{array}{cc|c} A_Q & B_Q C & B_Q D \\ A & \mathbf{0} & B \\ \hline C_Q & D_Q C & D_Q \end{array}\right] \\ &= \left[\begin{array}{cc|c} \multicolumn{2}{c|}{\widetilde{A}} & \widetilde{B} \\ \hline C_Q & D_Q C & D_Q D \end{array}\right] \end{aligned}$$

Finally we define

$$T_{zw}(z) = \left[\begin{array}{c|c} A_{zw} & B_{zw} \\ \hline C_{zw} & D_{zw} \end{array}\right]$$

Now, if the input to G_{zw} is $w(0)\dots w(k)$, $z(k)$ is given by

$$z(k) = t(k) + [\, C_Q \quad D_Q C \,] \, (\widetilde{A}^k x_0 + \sum_{p=0}^{k-1} \widetilde{A}^{k-1-p} \widetilde{B} w(p)) + D_Q$$

where $t(k)$ is output of T_{zw}. We can write this more compact as

$$z(k) = t(k) + h(k)\boldsymbol{q}$$

where

$$\boldsymbol{q}^T = [\, D_Q \quad C_Q \,] = [\, q_0 \dots q_{N-1} \,]$$

Also note that a realization of G_{zw} is given by

$$\left[\begin{array}{ccc|c} \widetilde{A} & \mathbf{0} & \widetilde{B} \\ \mathbf{0} & A_{zw} & B_{zw} \\ \hline C_Q & D_Q C_p & C_{zw} & D_{zw} + D_Q D \end{array}\right]$$

Constraints in Time-Domain

Recall that the composition of a convex function and a linear function is a convex function. Let c_{ik} for $1 \leq i \leq m$ be convex functions. Since the controlled output $z(k)$ is a linear function of the decision variables $\boldsymbol{q}$, constraints of the form

$$c_{ik}(z(k)) \leq 0$$

are convex in $\boldsymbol{q}$. If the step cost $l(z(k))$ is convex the finite horizon Dynamic MPC problem that we have defined in this chapter can therefore be solved as a static finite dimensional convex optimization problem.

If the Dynamic MPC problem is formulated with only time-domain constraints, it is sufficient to use a FIR- filter to parametrize $Q(z)$. We consider only $Q(z)$'s such that

$$Q(z) \in \text{span}\{1, z^{-1}, .., z^{-(N-1)}\} \subset \mathbb{RH}_\infty$$

If we choose N larger then the optimization horizon N_h, non of the terms $q_j z^{-j}$ with $N_h \leq j \leq N-1$ will affect the achieved optimal cost nor the constraint satisfaction. On the other hand, if we include frequency-domain constraints the length of the FIR-filter will in general affect the outcome.

Constraints in Frequency-Domain

Again let c be a convex function. A robustness constraint of the form

$$c(G_{zw}(z)) \leq W(z), \quad z = e^{i\omega}, \omega \in [-\pi, \pi]$$

can be well approximated by restricting the last inequality to a finite set of points ω_i. Each such constraint is convex in $\boldsymbol{q}$. Some important frequency-domain constraint can be treated without the need of gridding. Consider for example the Bounded Real lemma. It says that if A is stable then

$$||C(zI - A)^{-1}B + D||_\infty < \gamma$$

if and only if

$$\begin{bmatrix} A^T X A - X & A^T X B & C^T \\ B^T X A & B^T X B - \gamma I & D^T \\ C & D & -\gamma I \end{bmatrix} < 0$$

Note that this inequality is linear in X, C and D. We have noted that in the realization of G_{zw} the decision variables enter only in the C and D matrices, thus frequency domain peak bounds result in convex constraints.

6

Control of DC-DC Converters: A Case Study

6.1 Introduction

Switch-mode DC-DC converters is a class of power electronic circuits with a wide variety of applications, for example they can be found in all kinds of power supplies. Switch-mode converters are usually a good choice, as compared to non-switched converters, due to their high efficiency.

Power electronic circuits such as the switched mode DC-DC converters represent a good entry point for the investigation of the control design and performance benefits that can be brought by hybrid and optimal control techniques. Although these circuits may appear physically simple, high performance control design is nevertheless a challenging task. It is easy to find good mathematical models of the physical converters. Unfortunately, these models are highly nonlinear and they are parametrized by uncertain parameters. In addition to complicated models, the control design is further complicated by the fact that the control signal is physically constrained to an interval. Moreover, from a practical point of view it is necessary to limit system states as well, e.g. to impose current limitations. As a consequence some trade offs must be done.

Control design for switch mode power converters typically rely on the so called state space average model. This averaging methodology was outlined by Middlebrook *et al.* in [Middlebrook and Cuk, 1976], where the basic ideas of state-space averaging were introduced and the so-called small signal modeling was proposed for control design purposes. Detailed treatments of the averaging technique can be found in [Krein *et al.*, 1999] and [Lehman *et al.*, 1996].

The notion of averaging implies that only the dynamics of the DC

components of the circuit variables are taken into account. These are by definition quite slow compared to the switching period. On the other hand, the term small signal analysis describes the linearization of the resulting nonlinear state equations around the operating point. The average/small signal technique is convenient to use but it has several drawbacks. It has been noted, see [Fuad *et al.*, 2004] and references therein, that the classical average model need not capture the stability properties of the system, the average model can be stable even though the actual system is not. Also, a typical design specification is hard bounds on system variables, e.g. bounds on peak currents, since the state and the output ripple are underestimated in the average model such a specification must typically be handled by trial and error.

In practice, the design procedure for switched DC-DC converters is typically to first design a controller based on the averaged dynamics and then perform extensive numerical simulations to verify performance. Recently there has been a number of papers addressing the issues of stability, ripple estimation and harmonic analysis of switch mode power converters. For example, in [Almèr *et al.*, 2004; Geyer *et al.*, 2004] the authors perform stability analysis using sampled data techniques which take the switched nature of the system into account. The sampled data modeling results in a discrete time system which gives an exact description of the state at the switching instants. However, the resulting system is highly nonlinear and none of the classical design techniques can be applied. In the previous HYCON benchmark, see [Morari *et al.*, 2006] for initial definition of these problems, it has been shown that hybrid design techniques such as hybrid model predictive control and relaxed dynamic programming can be successfully employed for synthesis based on such sampled data models, see [Almer *et al.*, 2007; Beccuti *et al.*, 2007; Geyer *et al.*, 2004; Lincoln and Rantzer, 2002; Wernrud, 2006].

In this case-study we will present an extended version of the results reported in [Beccuti *et al.*, 2007; Almer *et al.*, 2007]. The problem we consider is control design of a fixed-frequency step-up converter, operating in continuous current mode, and a fixed-frequency step-down converter. The approach presented is to approximate the exact nonlinear sampled data model of the converters by a discrete time affine model and then apply the control design techniques presented in Chapter 4. Using this approach we can systematically incorporate both input and state constraints. Recently the work in [Beccuti *et al.*, 2007; Almer *et al.*, 2007] has been evaluated in real experiments. Some of these results will be discussed in this chapter, but a thorough description of the experiments and the results will appear in [Mariéthoz *et al.*, 2008b; Mariéthoz *et al.*, 2008a].

6.2 Physical Converter Models

The class of DC-DC converters that we will consider can be described by a nonlinear differential equation of the form

$$\dot{x}(t) = (A_0 + (A_1 - A_0)s_u(t))x(t) + B_0 + (B_1 - B_0)s_u(t)$$

where $x(t), B_i \in \mathbb{R}^n$, $A_i \in \mathbb{R}^{n\times n}$ and $i = 0, 1$. The value of the switching function $s_u(\cdot)$ at any time is either 0 or 1. It is the definition of the switching function that characterizes the converter. For fixed-frequency converters, the switching function is defined by the duty cycle u as follows: At fixed sample times kT_s the duty cycle $u(kT_s)$ takes a value in the interval $[0, 1]$, which is the ratio of the interval $[kT_s, (k+1)T_s]$ during which the switching function $s_u(\cdot)$ is equal to 1. In other words

$$s_u(t) = \begin{cases} 1, & t \in [kT_s, (k + u(kT_s))T_s) \\ 0, & t \in [(k + u(kT_s))T_s, (k + 1)T_s) \end{cases}$$

Therefore, the dynamic equation will switch between two affine systems

$$\dot{x}(t) = \begin{cases} A_1x(t) + B_1, & t \in [kT_s, (k + u(kT_s))T_s) \\ A_0x(t) + B_0, & t \in [(k + u(kT_s))T_s, (k + 1)T_s) \end{cases} \tag{6.1}$$

The states of the converter can be controlled by manipulating the duty cycle. Since, by definition, the duty cycle can only act at the beginning of each sampling interval the controlled system is inherently discrete. A discrete time model

$$x((k + 1)T_s) = \overline{\Phi}(u(kT_s))x(kT_s) + \overline{\Gamma}(u(kT_s)) \tag{6.2}$$

for the exact state propagation between time kT_s and $(k + 1)T_s$ can be found by integrating equation (6.1) over one period. The matrices in this model are given by

$$\begin{aligned} \overline{\Phi}(u(kT_s)) &= \overline{\Phi}_0(u(kT_s))\overline{\Phi}_1(u(kT_s)) \\ \overline{\Gamma}(u(kT_s)) &= \overline{\Phi}_0(u(kT_s))\overline{\Gamma}_1(u(kT_s)) + \overline{\Gamma}_0(u(kT_s)) \end{aligned}$$

where

$$\begin{aligned} \overline{\Phi}_1(u(kT_s)) &= e^{A_1T_su(kT_s)} \\ \overline{\Phi}_0(u(kT_s)) &= e^{A_0T_s(1-u(kT_s))} \\ \overline{\Gamma}_1(u(kT_s)) &= A_1^{-1}(e^{A_1T_su(kT_s)} - I)B_1 \\ \overline{\Gamma}_0(u(kT_s)) &= A_0^{-1}(e^{A_0T_s(1-u(kT_s))} - I)B_0 \end{aligned}$$

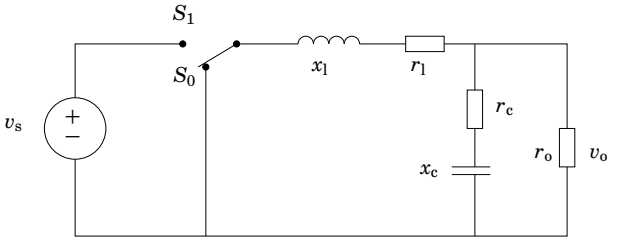


Figure 6.1 Synchronous buck converter.

The Step-Down Converter The topology of a step-down converter is shown in Figure 6.1. r_o denotes the ohmic output load, r_c the Equivalent Series Resistance (ESR) of the capacitor, r_ℓ denotes the internal resistance of the inductor, x_ℓ and x_c denotes the inductance and the capacitance respectively, and v_s denotes the input voltage. The state vector is define as $x(t) = [i_\ell(t)\ v_c(t)]^T$, where $i_\ell(t)$ is the current in the inductor and $v_c(t)$ is the voltage over the capacitor. The fixed switching period is denoted by T_s. During the time interval $[kT_s\ (k+u(kT_s))T_s)$ the switch in Figure 6.1 is in position S_1. In the remaining part of the interval, i.e. in $[(k+u(kT_s))T_s, (k+1)T_s))$, the switch is in position S_0. The corresponding matrices in equation (6.1) are given by

$$A_1 = A_0 = \begin{bmatrix} -\frac{1}{x_\ell}(r_\ell + \frac{r_o r_c}{r_o + r_c}) & -\frac{1}{x_\ell}\frac{r_o}{r_o + r_c} \\ \frac{1}{x_c}\frac{r_o}{r_o + r_c} & -\frac{1}{x_c}\frac{1}{r_o + r_c} \end{bmatrix}$$

$$B_1 = \begin{bmatrix} \frac{1}{x_\ell} \\ 0 \end{bmatrix}, \quad B_0 = 0.$$

The output voltage $v_o(t)$ across the load r_o is expressed as a function of the states through

$$v_o(t) = Cx(t) \quad \text{with} \quad C = \frac{r_o}{r_o + r_c} \begin{bmatrix} r_c & 1 \end{bmatrix}.$$

The circuit parameters that have been used in simulation and in the real experiments are given in Table 6.2. The values for r_o and v_s are nominal values.

x_{rmc}	100e-6	F
x_ℓ	2e-3	H
r_c	0.5	Ω
r_ℓ	0.25	Ω
r_o	50	Ω
v_s	40.0	V
T_s	5e-5	s

Table 6.1 Circuit parameter values for the buck converter

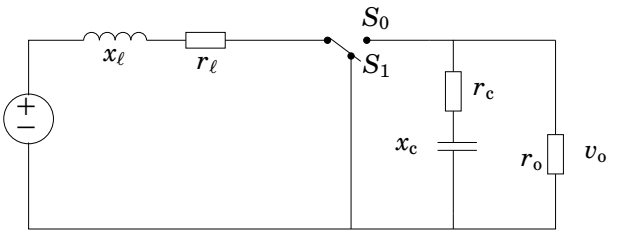


Figure 6.2 Topology of the boost (step-up) converter

The Step-up Converter The topology of a step-up converter is shown in Figure 6.2. The circuit parameters are the same as for the step-down topology. During the time interval $[kT_\mathrm{s}\ (k+u(kT_\mathrm{s}))T_\mathrm{s})$ the switch in Figure 6.2 is in position S_1. In the remaining part of the interval, i.e. in $[(k+u(kT_\mathrm{s}))T_\mathrm{s}, (k+1)T_\mathrm{s}))$, the switch is in position S_0. For the boost converter, the corresponding matrices in the model (6.1) are given by

$$
\begin{aligned}
A_1 &= \begin{bmatrix} -\dfrac{r_\ell}{x_\ell} & 0 \\ 0 & -\dfrac{1}{x_\mathrm{c}}\dfrac{1}{r_\mathrm{o}+r_\mathrm{c}} \end{bmatrix}, \\
A_0 &= \begin{bmatrix} -\dfrac{1}{x_\ell}(r_\ell + \dfrac{r_\mathrm{o}r_\mathrm{c}}{r_\mathrm{o}+r_\mathrm{c}}) & -\dfrac{1}{x_\ell}\dfrac{r_\mathrm{o}}{r_\mathrm{o}+r_\mathrm{c}} \\ \dfrac{1}{x_\mathrm{c}}\dfrac{r_\mathrm{o}}{r_\mathrm{o}+r_\mathrm{c}} & -\dfrac{1}{x_\mathrm{c}}\dfrac{1}{r_\mathrm{o}+r_\mathrm{c}} \end{bmatrix}, \\
B_1 = B_0 &= \begin{bmatrix} \dfrac{1}{x_\ell} \\ 0 \end{bmatrix} v_\mathrm{s}.
\end{aligned}
$$

The output voltage $v_o(t)$ across the load r_o is

$$v_o(t) = \begin{cases} C_1 x(t), & t \in [kT_s, (k + u(kT_s))T_s) \\ C_0 x(t), & t \in [(k + u(kT_s))T_s, (k+1)T_s) \end{cases}$$

with

$$C_1 = \begin{bmatrix} 0 & \dfrac{r_o}{r_o + r_c} \end{bmatrix}$$

and

$$C_0 = \begin{bmatrix} \dfrac{r_o r_c}{r_o + r_c} & \dfrac{r_o}{r_o + r_c} \end{bmatrix}.$$

The values of circuit parameters used in simulation and in the real experiments for the boost converter are shown in Table 6.2. The values for r_o and v_s are nominal values.

x_c	104e-6	F
x_ℓ	2e-3	H
r_c	2	Ω
r_ℓ	1.0	Ω
r_o	200	Ω
v_s	20.0	V
T_s	5e-5	s

Table 6.2 Circuit parameter values for the boost converter.

6.3 Modeling for Control Design

As mentioned in the introduction, we will design our controllers using the methods developed in Chapter 4. Since those methods require that the system dynamics is affine we need to approximate the converter dynamics (6.2) with such a system. The modeling technique presented below, which may be refereed to as robust affine approximation, is proposed in order to take into account, already at the modeling stage, the switched nature of the converters and the fact that the converters are usually parametrized by unknown parameters.

Robust Affine Approximation

Consider again the state update equation

$$x((k+1)T_s) = \overline{\Phi}(u(kT_s), r_o)x(kT_s) + \overline{\Gamma}(u(kT_s), r_o) \tag{6.3}$$

where

$$\overline{\Phi}(u, r_o) = \overline{\Phi}_0(u, r_o)\overline{\Phi}_1(u, r_o)$$
$$\overline{\Gamma}(u, r_o) = \overline{\Phi}_0(u, r_o)\overline{\Gamma}_1(u, r_o) + \overline{\Gamma}_0(u, r_o)$$

are the matrices in (6.2). The load parameter r_o has been appended to emphasize that the matrices depend on the load.

Our approach is to approximate the nonlinear dynamics by a constant affine system

$$x((k+1)T_s) = \Phi x(kT_s) + \Gamma u(kT_s) + \nu \tag{6.4}$$

When the model (6.2) is approximated with (6.4) the largest pointwise error can be expressed as

$$J = \sup ||\Phi x + \Gamma u + \nu - (\Phi(u, r_o)x + \Gamma(u, r_o))||$$

where the supremum is taken over $(x, u, r_o) \in \mathbb{X} \times \mathbb{U} \times \mathbb{L}$, where $\mathbb{X}$ is the set of states on which the model should be approximated. $\mathbb{L}$ is the set of values that the load can assume. Naturally, we would like to minimize J. The robust approximation problem is to compute

$$\min J(\Phi, \Gamma, \nu) \tag{6.5}$$

over (Φ, Γ, ν). Our ability to solve this problem depends on the choice of norm and the description of the set $\mathbb{X} \times \mathbb{U} \times \mathbb{L}$, the candidates are those that correspond to a finite dimensional convex optimization problem. For the purpose of this chapter we shall consider a simple choice. To be able to write this problem in a familiar form we introduce the Kronecker product of two matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{mp \times nq}$$

also the vectorization operation of a matrix $A \in \mathbb{R}^{m \times n}$ is defined as

$$\text{vec}(A) = \text{vec}(\begin{bmatrix} a_1 & \dots & a_n \end{bmatrix}) = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \in \mathbb{R}^{mn}$$

i.e. the columns in A are stacked into one column vector.

Define a finite grid of points $G \subset \mathbb{X} \times \mathbb{U} \times \mathbb{L}$. For any $g = [x_g^T \quad u_g \quad r_g] \in G$ define

$$b(g) = \Phi(u_g, r_g)x_g + \Gamma(u_g, r_g), \text{ and } A(g) = g \otimes I$$

If we also take $y = \text{vec}([\Phi \quad \Gamma \quad \nu])$, the approximation problem becomes

$$\min_y \max_{g \in G} ||A(g)y - b(g)||$$

which is the same as

$$\begin{aligned} &\min_{y,t} \quad t \\ &||A(g)y - b(g)|| \leq t, \quad \forall g \in G \end{aligned}$$

if the norm is either $||\cdot||_\infty$ or $||\cdot||_1$ this is an LP, if norm is $||\cdot||_2$ the problem is a second order cone problem. In any case, it is an easily solvable finite dimensional convex optimization problem.

A small example is provided to verify that it might be useful to consider the proposed approximation.

EXAMPLE 6.1
We consider the problem to approximate the buck converter dynamics with an affine system. First consider an approximation based on the assumption of constant load $r_o = 50$. The problem (6.5) was solved, using the 2-norm, on a grid given by

$$G = \{0.2j, j \in [-1, 13]\} \times \{j, j \in [0, 30]\} \times \{0.2j, j \in [0, 5]\} \times \{50\}.$$

For a given input sequence $u(k)$, $0 \leq k \leq 99$ and initial conditions $x(0) = x^a(0) = [0 \ 0]^T$ let $x^a(k)$ be the solution to (6.4). Let $x(k)$ be the solution to (6.3) when the true load $r_o(k)$ is given by a square wave

$$r_o(k) = \begin{cases} 100, & 21 \leq k \leq 40 \text{ or } 61 \leq 80 \\ 50, & \text{else} \end{cases}$$

As a measure of how good the model approximation is we use the following

$$\hat{F}(j) = \mathbb{E}F(x_j(k) - x_j^a(k), u(k))$$

where $\mathbb{E}(\cdot)$ is the expectation operator and

$$F(x_j(k) - x_j^a(k), u(k)) = \sqrt{\frac{1}{N}\sum_1^N |x_j(k) - x_j^a(k)|^2}$$

and $u(k)$ is uniformly distributed in $[0,1]$ for each k. Sampling F 10000 times gives the estimates

$$\hat{F}(1) \approx 0.2922$$
$$\hat{F}(2) \approx 0.7801$$

The mean value of $F(x_1(k),u(k))$ was approximately 1 and the mean of $F(x_2(k),u(k))$ was approximately 22. Consider the same setup except that the grid is extended by adding an extra point $r_o = 100$ in the load direction

$$G = \{0.2j, j \in [-1,13]\} \times \{j, j \in [0,30]\} \times \{0.2j, j \in [0,5]\} \times \{50,100\}.$$

In this case an error estimate is given by

$$\hat{F}(1) \approx 0.1624$$
$$\hat{F}(2) \approx 0.5090$$

Thus, the expected error in the first coordinate is reduced by approximately 44% and in the second coordinat by 35%. The mean values of $F(x_1(k),u(k))$ and of $F(x_2(k),u(k))$ were approximately the same as before. □

6.4 Case Study 1: Control Design for the Step-Down Converter

The main control objective is to regulate the DC component of the output voltage to its reference $v_{\mathrm{o,ref}}$. The duty cycle should be kept constant during steady-state operation in order to avoid undesired phenomena such as subharmonic oscillations. The closed loop should be designed so that regulation can be maintained in the presence of both measurable voltage source variations and unmeasurable output load disturbances. Moreover, for the step-down converter we have imposed a current limit of $i_{\ell,max} = 2.5$ A, the hard constraint on the duty cycle, i.e. $0 \leq u(kT_{\mathrm{s}}) \leq 1$, must of course be satisfied too. The control problem is further complicated by the uncertainty and variation of the component values on which the controller synthesis process is based on.

The following tests were used to evaluate the closed loop performance.

1. General performance: Given the initial state $x(0) = [0,\ 0]^T$ and $v_{\mathrm{s}} = 50$ V we test the speed of the closed loop by setting the reference to 20 V, 25 V and 30 V.

2. Sensitivity to load variations: the load is often subject to large variations during the operation, the closed loop must be robust against load transients. The nominal load is $r_o = 50\ \Omega$, for each setpoint in the first test we double the load when stationary conditions have been reached and reset it again.
3. Sensitivity to line variations: the supply voltage is subject to large variations during the operation, the closed loop must be robust against line transients. Robustness is tested by applying a drop in the nominal supply voltage $v_s = 50$ V to 35 V and then reset it again.
4. Robustness to parameter variation is tested by varying the capacitor during closed loop operation.

Control Design

For the step down converter, the $\overline{\Phi}$-matrix in the nonlinear model (6.2) is independent of the duty cycle and the $\overline{\Gamma}$-matrix is essentially linear in the duty cycle. Thus, the model (6.2) can be well approximated with a linear system

$$x((k+1)T_s) = \Phi x(kT_s) + \Gamma u(kT_s) + \nu \tag{6.6}$$

The Relaxed Dynamic Programming procedure of Chapter 4 was used to synthesize a controller $\mu(x)$ such that the total cost $V = \sum_k l(x(kT_s), u(kT_s))$ was approximately minimized under an additional constraint on the inductor current, $x_1(kT_s) \leq 2.5$ A and on the duty cycle $0 \leq u(kT_s) \leq 1$. The step cost was chosen as

$$l(x,u) = q_1|v_o(kT_s) - v_{o,\mathrm{ref}}| + q_2|u(kT_s) - u((k-1)T_s)|$$

where q_1 and q_2 are positive weights. Note that the step cost can be represented as a max of linear functions

$$l(x,u) = \max_{q \in Q} q^T e(x,u)$$

as required in Chapter 4

The penalty on consecutive control values was introduced to force the duty cycle to become constant when $v_o(kT_s)$ reached the output voltage reference. Thus, an extra state $x_e(kT_s) = u((k-1)T_s)$ was introduced.

Due to the fact that on the real plant there is one sample delay between measurement and actuation the model used to compute the controller was

$$\begin{bmatrix} x((k+1)T_s) \\ x_e((k+1)T_s) \end{bmatrix} = \begin{bmatrix} \Phi & \Gamma \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(kT_s) \\ x_e(kT_s) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(kT_s) + \begin{bmatrix} \nu \\ 0 \end{bmatrix}$$

A stationary approximate value function $\hat{V}^{26}$ was found after 26 relaxed value iterations, with guaranteed bounds

$$0.6V^* \leq \hat{V}^{26} \leq 2.6V^*$$

where V^* is the optimal total cost function. The approximate value function was given by a max of 136 linear functions. By applying the controller reduction algorithm in Chapter 4 the controller table could be reduced to a size of 106 entries.

The errors introduced by the model approximation were handled by an outer integrator loop that was used to adjust the voltage reference. The integrator was activated only when the output voltage was sufficiently close to its reference value: If the test

$$\frac{|v_{\mathrm{o,ref}} - v_{\mathrm{o}}(kT_{\mathrm{s}})|}{v_{\mathrm{o,ref}}} \leq 5\%$$

was true the integrator state I was updated to

$$I((k+1)T_{\mathrm{s}}) = I(kT_{\mathrm{s}}) + k_I(v_{\mathrm{o,ref}} - v_{\mathrm{o}}(kT_{\mathrm{s}}))$$

otherwise no update was made. This strategy was employed to ensure that the integrator only was used to correct small gain errors.

Finally, the measurable source voltage was used in a feedforward loop to adjust the input gain according to

$$\frac{v_{\mathrm{s,nom}}}{v_{\mathrm{s}}(kT_{\mathrm{s}})}\mu(x(kT_{\mathrm{s}}))$$

Simulation Results for the Buck Converter

Figures 6.3-6.6 show the simulation results for the buck converter. The simulations were performed with one full sampling period delay.

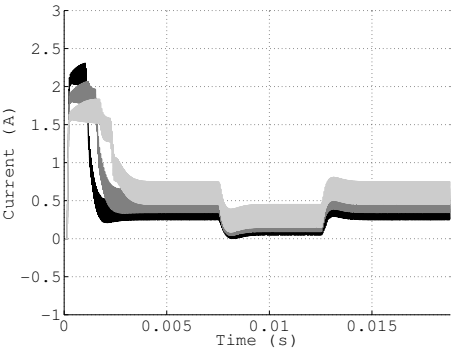
In Figure 6.3 the system is started from zero initial condition and three different voltage references are applied. For each transient response a response to a step in the load from the nominal value 50 Ω up to 100 Ω and then back again is also shown. Figure 6.4 shows the transient and the response to a step in the source voltage from the nominal 50 V down to 35 V and then back again. Figure 6.5 shows how the transient varies for different values of the capacitance. The nominal capacitance is $x_{\mathrm{c,nom}} = 100\ \mu$F, corresponding to the black curve. The gray curve corresponds to a value on the capacitance of $0.5x_{\mathrm{c,nom}}$ and finally the light-gray curve corresponds to $2x_{\mathrm{c,nom}}$. The simulation in Figure 6.6 shows the response

to a step in the source voltage from the nominal 50 V down to 35 V and then back again for a design with no computational delay, i.e. the model

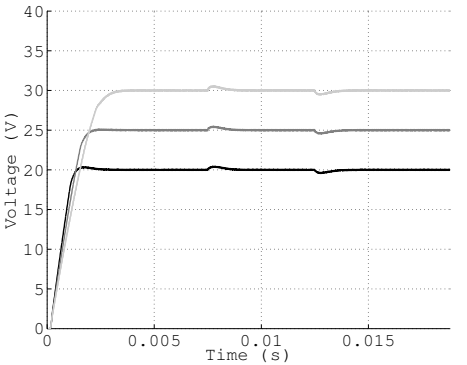
$$\begin{bmatrix} x((k+1)T_s) \\ x_e((k+1)T_s) \end{bmatrix} = \begin{bmatrix} \Phi & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(kT_s) \\ x_e(kT_s) \end{bmatrix} + \begin{bmatrix} \Gamma \\ 1 \end{bmatrix} u(kT_s) + \begin{bmatrix} \nu \\ 0 \end{bmatrix}$$

was used to compute the controller. The response is similar to that in Figure 6.4 and shows that the control design is robust to computational delays.

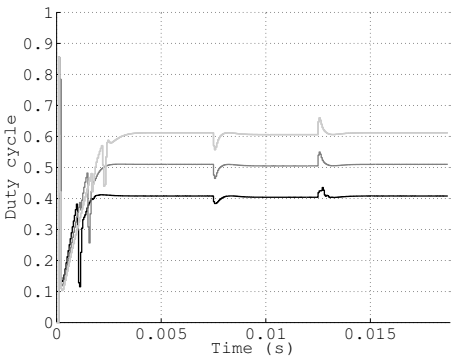
The simulation results show excellent performance. The success of the outlined synthesis procedure can largely be explained by the fact that we approximate the infinite horizon optimal controller. As such, our approximation inherits robustness properties from the stationary optimal controller. So even though we use a relatively coarse model during synthesis the closed loop turns out to have good performance.



(a) Inductor current

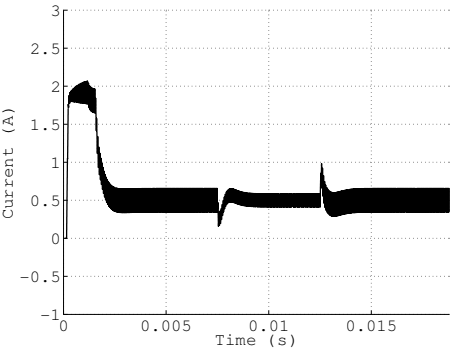


(b) Output voltage

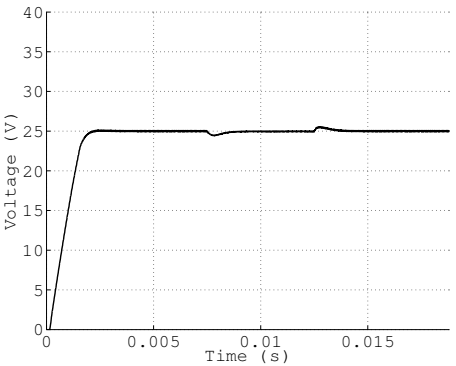


(c) Duty cycle

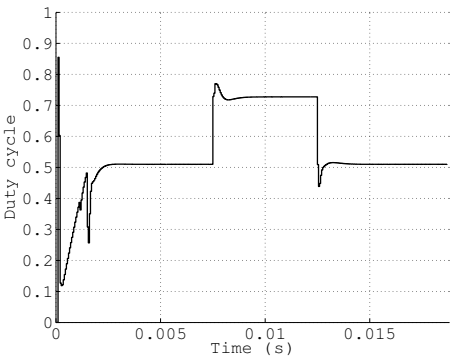
Figure 6.3 Step response and response to a step in the load; inductor current (top), output voltage (middle) and duty cycle (bottom). We let $v_s = 50$ V, $v_{ref} = 20, 25, 30$ V, and there is a step in the load from $r_o = 50\ \Omega$ to $r_o = 100\ \Omega$ and then back again.



(a) Inductor current

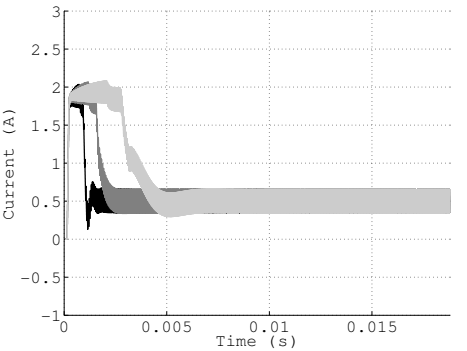


(b) Output voltage

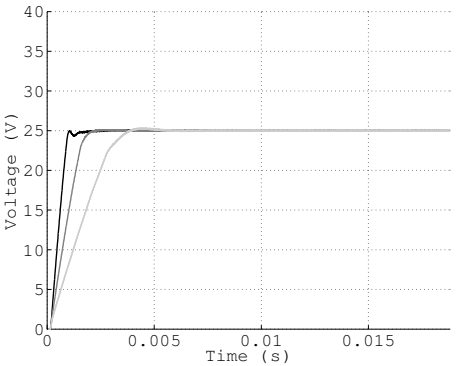


(c) Duty cycle

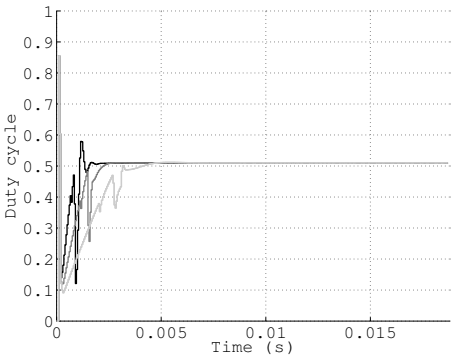
Figure 6.4 Step response and response to a step in the source; inductor current (top), output voltage (middle) and duty cycle (bottom).



(a) Inductor current



(b) Output voltage



(c) Duty cycle

Figure 6.5 Step response for capacitance $0.5x_{c,\mathrm{nom}} = 50\ \mu\mathrm{F}$ (light-gray), $x_{\mathrm{c,nom}} = 100\ \mu\mathrm{F}$ (black) and $2x_{\mathrm{c,nom}} = 200\ \mu\mathrm{F}$ (gray); inductor current (top), output voltage (middle) and duty cycle (bottom).

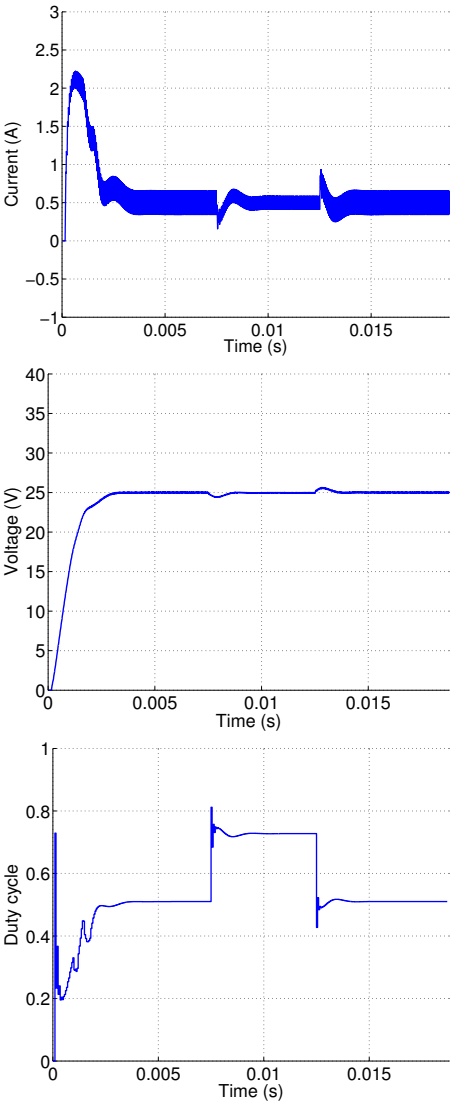


Figure 6.6 Step response and response to a step in the source for the case with zero computational delay; inductor current (top), output voltage (middle) and duty cycle (bottom).

6.5 Case Study 2: Control Design for the Step-Up Converter

The main control objective is to regulate the DC component of the output voltage to its reference $v_{o,ref}$. The duty cycle should be kept constant during steady-state operation in order to avoid undesired phenomena such as subharmonic oscillations. The closed loop should be designed so that regulation can be maintained despite measurable voltage source variations and unmeasurable output load disturbances. Moreover, for the step-up converter we have imposed a current limit of $i_{\ell,max} = 2.5$ A, the hard constraint on the duty cycle, i.e. $0 \leq u(kT_s) \leq 1$, must of course be satisfied too. The control problem is further complicated by the uncertainty and variation of the component values on which the controller synthesis process is based and by the non-minimum phase behaviour of the output voltage with respect to the duty cycle.

The following tests have been used to evaluate the closed loop performance.

1. Load transient: The controller is switched on and steers the output voltage to its reference, thereafter the load is subjected to a step-down variation of 50% i.e. r_o is set to 100 Ω, and is then restored back to its original value. The scenario was run for 3 different supply voltages of 15, 20 and 25 V.

2. Robustness to capacitor variations: The controller steers the output voltage to its reference. The scenario was run for a supply voltage of 20 V and with 3 different values of x_c equal to 50, 100 and 200 μF.

3. Line transient: First $v_s = 25$ V and the controller steers the output voltage to its reference, thereafter the supply is decreased to 15 V and then restored to its original value.

Control Design

The design procedure used for the step-down converter was also used for the step-up converter. For the step-up converter, however, the synthesis was more difficult for several reasons.

First, the boost converter dynamics has non-minimum phase behaviour from the duty cycle to the output voltage, which was captured by our model approximation. As a consequence, the convergence rate of value iteration was slower than for the step-down converter. This, in turn, resulted in a more complex value function.

Also, even though our model approximation did capture the non-minimum phase behaviour of the nonlinear model, it was not as good as for the step-down converter.

Despite these problems we were able to find a good design, after considerable tuning of the design parameters. The step cost was chosen as

$$l(x,u) = q_1|v_o - v_{ref}| + q_2|u(kT_s) - u((k-1)T_s)|$$

with $q_1/q_2 = 1/5$, compared to $q_1/q_2 = 1/2$ which was used for the buck. The relaxed value iteration algorithm was executed 38 times resulting in a stationary approximation which satisfied

$$0.4V^* \leq \hat{V}^{38} \leq 3.2V^*$$

The approximate value function were given by a max of 166 linear functions By applying the controller reduction algorithm in Chapter 4 the controller table could be reduced to a size of 121 entries.

Experimental Setup

The controller design described in the previous section has been evaluated in real experiment. The design and configuration of the converter hardware will not be discussed here[1], but see [Mariéthoz *et al.*, 2008b; Mariéthoz *et al.*, 2008a] for a description.

The controller was implemented on a DSP control board consisting of a 16 bit 600 MHz fixed point processor from Analog Device Blackfin. The realtime process can be summarized as:

1. The controller samples the converter state and performs analog to digital conversion.
2. Execute the control algorithm.
3. Actuate at the end of the sampling interval.

The employed sampling frequency was 20 kHz, i.e. the sampling interval was 50 μs. It took about 5 μs to sample the state and perform A/D-conversion. The computation time required by the control algorithm was about 15 μs.

Evaluation of Boost Converter Design

Figures 6.7–6.10 show the simulation and experimental results for the boost converter. Each simulation was performed with one full sampling period delay to account for the measurement, conversion and controller computation times.

Figures 6.7(a)–6.7(f) show the transient and the response to a step in the load from the nominal value 200 Ω down to 100 Ω and then back

[1]The hardware has been built and is maintained by the control group at ETH, Zurich

again for 3 different voltage supply values. The black curve corresponds to 15 V, gray corresponds to 20 V and finally 25 V corresponds to the light-gray curve.

Figures 6.9(a)–6.9(f) show the transient for different values of the capacitance. The gray curve corresponds to the nominal capacitance $x_{\mathrm{c,nom}} = 100\ \mu\mathrm{F}$, the black corresponds to $0.5x_{\mathrm{c,nom}}$ and the light-gray corresponds to $2x_{\mathrm{c,nom}}$.

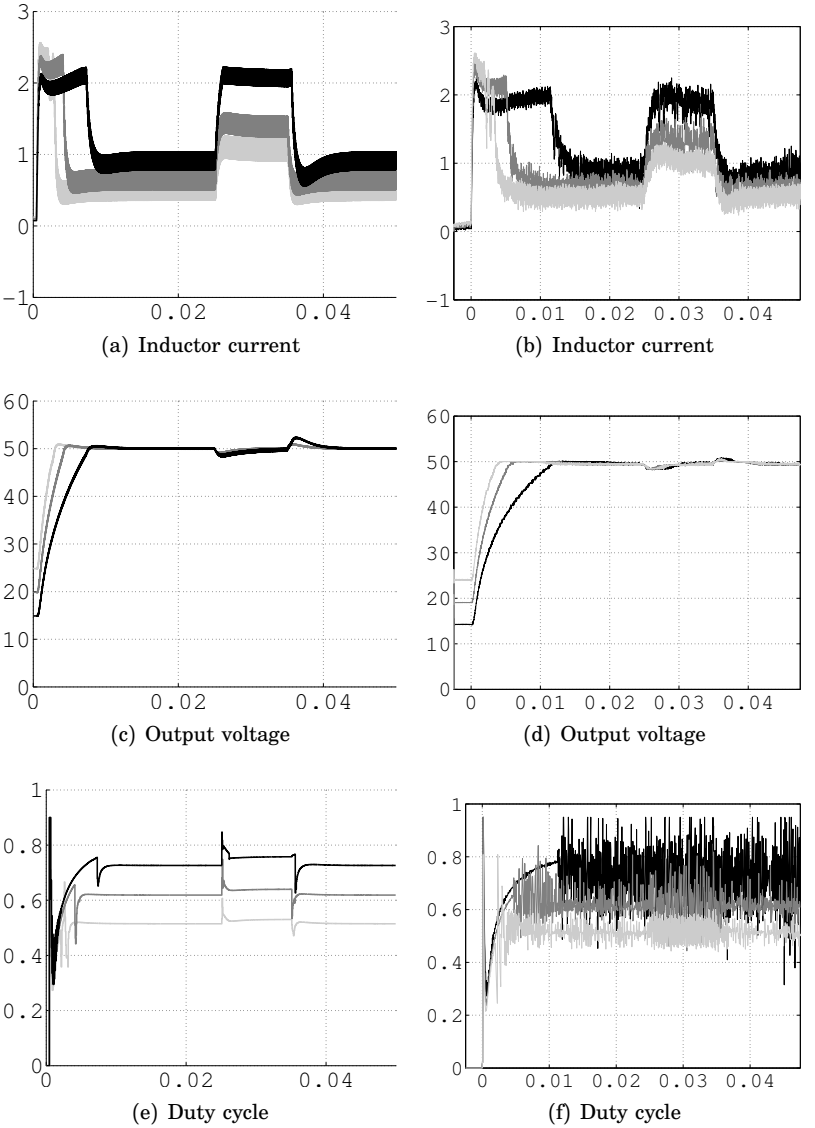
Figures 6.8(a)–6.8(f) show the response to a step in the source voltage from 25 V down to 15 V and then back again, see Figure 6.11.

Finally, the same scenario, but with a zero delay assumption in the controller synthesis phase, is shown in Figure 6.10. The response is similar to left column in Figure 6.8 and shows that the control design is robust to computational delays.

Just as for the buck, the simulation results show excellent performance.

Overall, the experimental results also agrees qualitatively with the simulated results. For example, good transient response and voltage tracking were achieved in all scenarios. Moreover, the current limit is respected in each test.

The undesirable scattering in the duty cycle is believed to be due to noise, parametric uncertainty, quantization errors related to the A/D conversion. Moreover, the fact that the controller was implemented on a fixed point platform introduces a precision loss in the controller tables due to quantization. Extensive simulations in fixed point indicate that the scattering in the duty cycle may also be caused by that. See Figure 6.12 for a simulation when the controller is implemented in fixed point.



(a) Inductor current (b) Inductor current

(c) Output voltage (d) Output voltage

(e) Duty cycle (f) Duty cycle

Figure 6.7 Step response and response to a step in the load. Simulation results left and experimental results right.

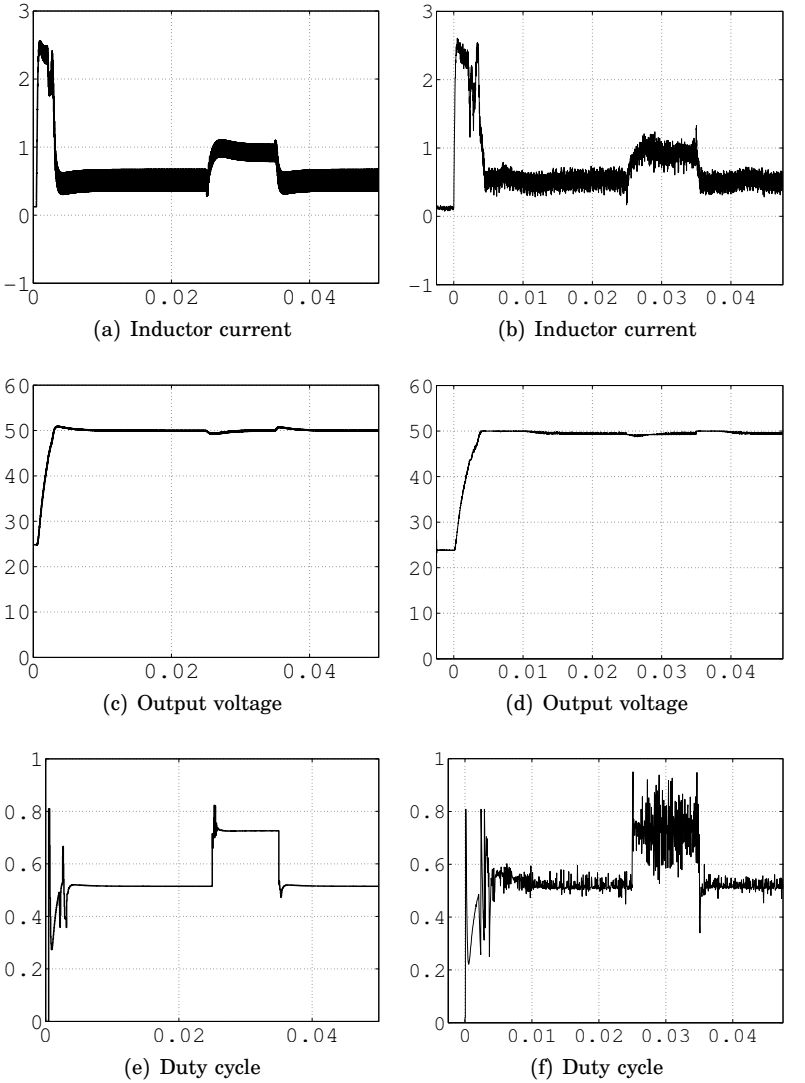


Figure 6.8 Result for a steps in the voltage supply: The left column shows simulations and the right column shows experimental results.

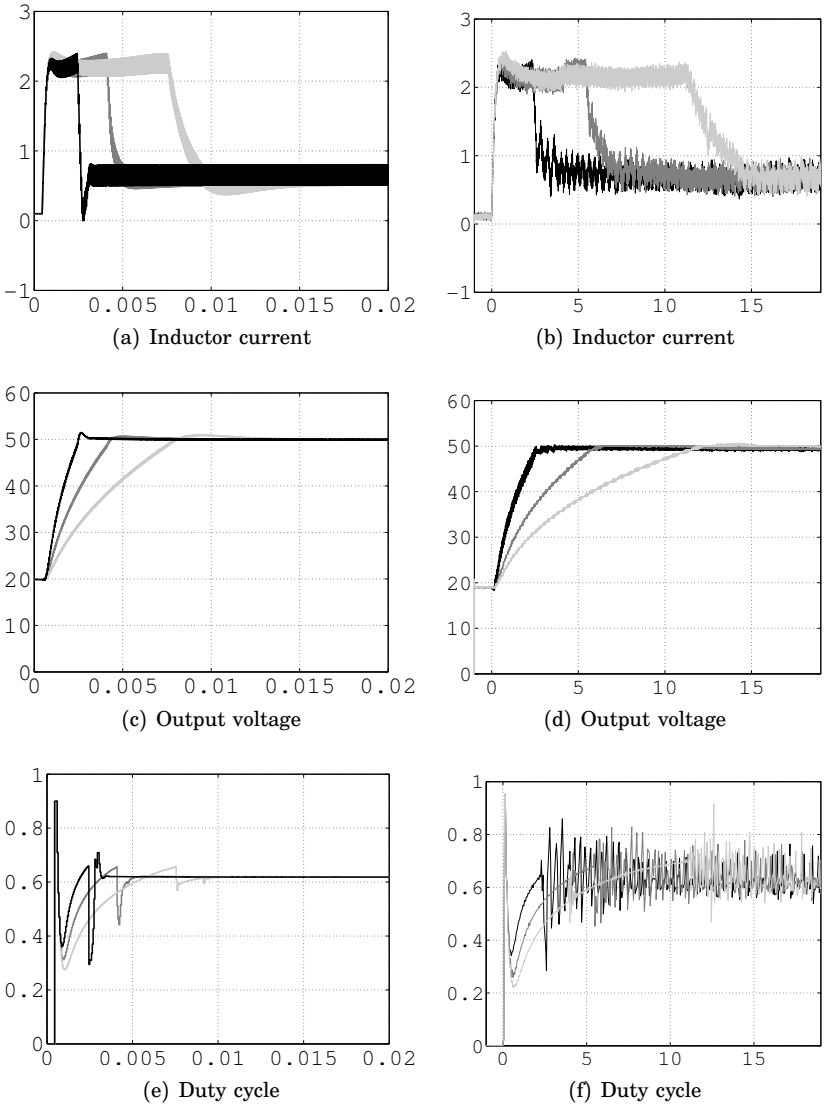
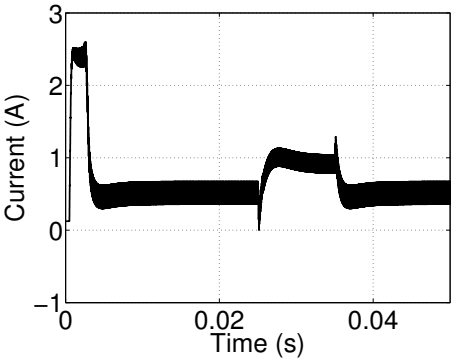
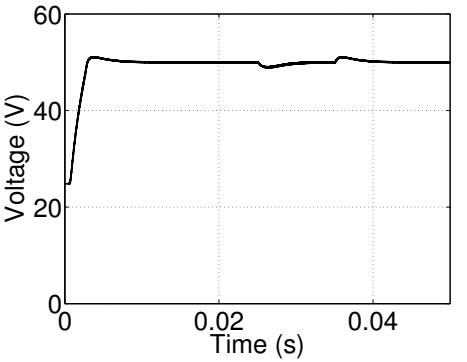


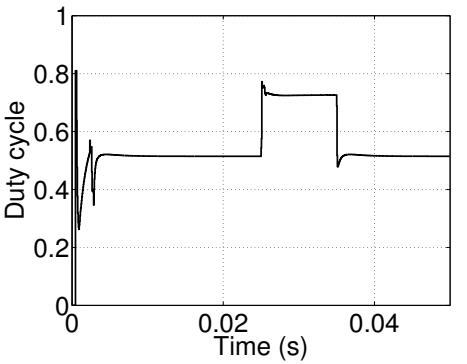
Figure 6.9 Results for variations in the capacitor. The left column shows results from simulations, right column shows results from experiments.



(a) Inductor current

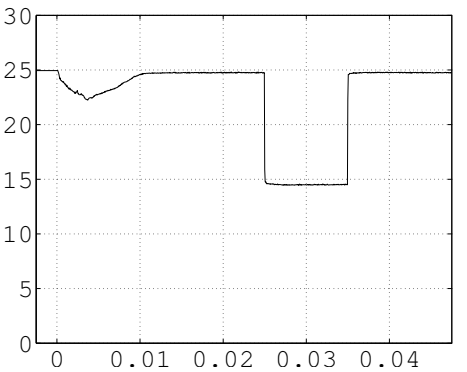


(b) Output voltage



(c) Duty cycle

Figure 6.10 Simulation results for steps in the voltage supply with zero delay.



(a) Supply voltage

Figure 6.11 Steps in the voltage supply used in experiments.

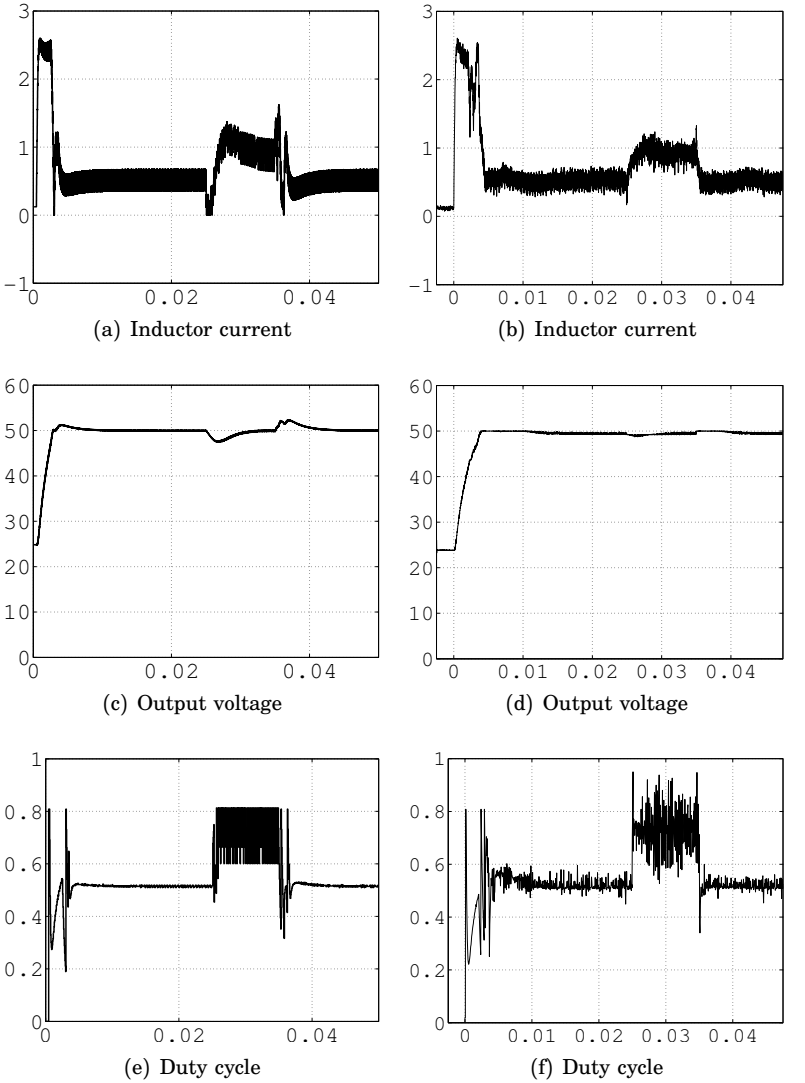


Figure 6.12 The left column shows simulations with quantized lookup tables, the right column shows data from the real experiment.

6.6 Summary and Concluding Remarks

This chapter has presented a case study for the synthesis of high performance controllers for fixed frequency boost and buck DC–DC converters. The circuits used present a number of challenges. First, the switched nature of the system dynamics makes them highly nonlinear. The nonminimum phase behaviour for the boost converter and the input and state constraints additionally complicate the controller design process.

A controller design based on relaxed dynamic programming has been presented. The controllers have been evaluated in simulations for both the boost and the buck converter. In addition, the design for the boost converter has been evaluated in experiments.

The simulation results showed that the proposed design was feasible for both types of converters. In particular all the design specifications were met.

The experimental results for the boost converter agreed qualitatively with the simulations. It is believed that the undesirable scattering in the duty cycle is a consequence of quantization errors and noise.

References

Almer, S., H. Fujioka, U. Jönsson, C.-Y. Kao, D. Patino, P. Riedinger, T. Geyer, A. Beccuti, G. Papafotiou, M. Morari, A. Wernrud, and A. Rantzer (2007): "Hybrid control techniques for switched-mode DC-DC converters, part I: The step-down topology." In *American Control Conference*. New York City, USA.

Almèr, S., U. Jönsson, C.-Y. Kao, and J. Mari (2004): "Global stability analysis of DC-DC converters using sampled-data modeling." In *Proceedings of the American Control Conference*, pp. 4549–4554. Boston, MA, USA.

Bardi, M. and I. Capuzzo-Dolcetta (1997): *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Birkhäuser.

Beard, R., G. N. Saridis, and J. T. Wen (1998): "Approximate solutions to the time-invariant Hamilton-Jacobi-Bellman equation." *Journal of Optimization Theory And Applications*, **96:3**, pp. 589–626.

Beccuti, A., G. Papafotiou, M. Morari, S. Almer, H. Fujioka, U. Jönsson, C.-Y. Kao, A. Wernrud, A. Rantzer, M. Baja, H. Cormerais, and J. Buisson (2007): "Hybrid control techniques for switched-mode DC-DC converters, part II: The step-up topology." In *American Control Conference*. New York City, USA.

Beeler, S. C., H. T. Tran, and H. T. Banks (2000): "Feedback control methodologies." *J. of Optimization Theory and Application*, **107:1**, pp. 1–33.

Bellman, R. (1957): *Dynamic Programming*. Princeton University Press.

Bemporad, A., F. Borrelli, and M. Morari (2000): "The explicit solution of constrained LP-based receding horizon control." In *IEEE Conference on Decision and Control*. Sydney, Australia.

Bertsekas, D. P. (2000): *Dynamic Programming and Optimal Control: Vol. 1*. Athena Scientific.

Boyd, S. and C. H. Barratt (1991): *Linear Controller Design: Limits of Performance*. Prentice Hall.

Boyd, S. P., V. Balakrishnan, C. Barratt, N. Khraishi, X. Li, D. Meyer, and S. Norman (1988): "A new CAD method and associated architectures for linear controllers." *IEEE Trans. on Automatic Control*, **33**, pp. 268–283.

Cesari, L. (1983): *Optimization-Theory and Applications*. Springer-Verlag.

CLP (2007): "Coin-or linear program solver, clp-160." http://www.coin-or.org/Clp/index.html.

Fleming, W. H. and H. M. Soner (1993): *Controlled Markov Processes and Viscosity Solutions*. Springer, Applications of Mathematics.

Fuad, Y., W. L. de Koning, and J. van der Woude (2004): "On the stability of the pulsewidth-modulated cuk converter." *IEEE Trans. Circuits Syst.*, **2:51**, pp. 412–420.

Garcia, C. E., D. Prett, and M. Morari (1989): "Model predictive control: Theory and practice - a survey." *Automatica*, **25**, pp. 335–348.

Garrard, W. (1969): "Additional results on suboptimal feedback control of nonlinear systems." *Int. J. of Control*, **10:6**, pp. 657–663.

Garrard, W. L. and J. Jordan (1977): "Design of nonlinear automatic flight control systems." *Automatica*, **13**, pp. 497–505.

Geyer, T., G. Papafotiou, and M. Morari (2004): "On the Optimal Control of Switch-mode DC-DC Converters." *Hybrid Systems: Computation and Control*, **2993**, March, pp. 342–356.

Goulart, P., E. Kerrigan, and J. Maciejowski (2006): "Optimization over state feedback policies for robust control with constraints." *Automatica*, **42:4**, pp. 523–533.

Grüne, L. (1997): "An adaptive grid scheme for the discrete Hamilton-Jacobi-Bellman equation." *Numerische Mathematik*, **75:3**, pp. 319–337.

Grüne, L. and A. Rantzer (2006): "Suboptimality estimates for receding horizon controllers." In *The 17th International Symposium on Mathematical Theory of Networks and Systems*. Kyoto,Japan.

Gutman, P. O. and M. Cwikel (1986): "Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states." *IEEE Transactions on Automatic Control*, **31**, pp. 373–376.

Heriksson, D. (2006): *Resource-Constrained Embedded Control and Computing Systems*. PhD thesis ISRN LUTFD2/TFRT--1074--SE, Department of Automatic Control, Lund Institute of Technology, Sweden.

Huang, Y. and W. M. Lu (1996): "Nonlinear optimal control: Alternatives to Hamilton-Jacobi equation." *In Proc. IEEE Conf. on Decision and Control*.

Kleinman, D. L. (1968): "On an iterative technique for Riccati equation computations." *IEEE Transactions on Automatic Control*, **13**, pp. 114–115.

Koutsoukos, X. D. and P. J. Antsaklis (2002): "Design of stabilizing switching control laws for discrete and continuous-time linear systems using piecewise-linear Lyapunov functions." *International Journal of Control*, **75:12**, pp. 932–945.

Krein, P., J. Bentsman, R. M. Bass, and B. L. Lesieutre (1999): "On the use of averaging for the analysis of power electronic systems." *IEEE Trans. Power Electron*, **5:2**, pp. 182–190.

Lasserre, J. B. (2002): "Semidefinite programming vs. lp relaxations for polynomial programming." *Math. Oper. Res.*, **27:2**, pp. 347–360.

Leake, R. J. and R.-W. Liu (1967): "Construction of suboptimal control sequences." *SIAM J. Control and Optimization*, **5:1**, pp. 54–63.

Lehman, B. A., , and R. M. Bass (1996): "Extensions of averaging theory for power electronic systems." *IEEE Trans. Power Electron.*, **11:4**, pp. 542–553.

Leitmann, G. (1981): *The Calculus of Variations and Optimal Control*. Plenum Press.

Lincoln, B. (2003): *Dynamic Programming and Time-Varying Delay Systems*. PhD thesis ISRN LUTFD2/TFRT--1067--SE, Department of Automatic Control, Lund Institute of Technology, Sweden.

Lincoln, B. and A. Rantzer (2002): "Suboptimal dynamic programming with error bounds." In *Proc. 41st IEEE Conference on Decision and Control*.

Lincoln, B. and A. Rantzer (2006): "Relaxing dynamic programming." *IEEE Transactions on Automatic Control*, **51:8**, pp. 1249–1260.

Löfberg, J. (2003): "Approximations of closed-loop minimax mpc." In *IEEE Conference on Decision and Contro*. Hawaii, USA.

Luenberger, D. G. (1984): *Linear and Non-linear Programming*. Prentice Hall.

Lukes, D. L. (1969): "Optimal regulation of nonlinear dynamical systems." *SIAM Journal on Control*, **7:1**, pp. 75–100.

Löfberg, J. (2004): "Yalmip : A toolbox for modeling and optimization in MATLAB." In *Proceedings of the CACSD Conference*. Taipei, Taiwan.

Maciejowski, J. M. (2002): *Predictive control: with constraints*. Prentice Hall.

Mariéthoz, S., S. Almér, A. Beccuti, D. Patino, A. Wernrud, T. Geyer, H. Fujioka, U. Jönsson, C.-Y. Kao, M. Morari, G. Papafotiou, A. Rantzer, and P. Riedinger (2008a): "Evaluation of four hybrid control techniques for the synchronous step down buck DC-DC converter." Submitted.

Mariéthoz, S., S. Almér, B. Mihai, A. G. Beccuti, A. Wernrud, H. Fujioka, U. Jönsson, C.-Y. Kao, H. Cormerais, J. Buisson, G. Papafotiou, M. Morari, and A. Rantzer (2008b): "Comparative assessment of hybrid control techniques for the boost DC-DC converter." Submitted.

Markman, J. and I. Katz (2000): "An iterative algorithm for solving hamilton-jacobi equations." *SIAM J. Sci. Comput.*, **22**, pp. 312–329.

Mayne, D. Q., J. B. Rawlings, C. V. Rao, and P. O. Scokaert (2000): "Constrained model predictive control: Stability and optimality." *Automatica*, **36:6**, pp. 789–814.

Middlebrook, R. D. and S. Cuk (1976): "A general unified approach to modeling switching-converter power stages." In *Proc. IEEE Power Electronics Specialists Conf. (PESC)*, pp. 18–34.

Morari, M., J. Buisson, B. de Schutter, and G. Papafotiou (2006): "Report on the assessment of hybrid control methods for electric energy management problems." Technical Report IST contract number 511368. HYCON Deliverable.

Mårtensson, K., A. Wernrud (2008): "Dynamic Model Predictive Control" To appear at the 17th IFAC World Congress.

Nishikawa, Y., N. S. and H. Itakura (1971): "A method for suboptimal design of nonlinear feedback systems." *Automatica*, **7**, pp. 703–712.

Parrilo, P. A. (2003): "Semidefinite programming relaxations for semialgebraic problems." *Mathematical Programming Ser. B*, **96:2**, pp. 293–320.

Prajna, S., A. Papachristodoulou, and P. Parrilo (2002): "Introducing SOSTOOLS a general purpose sum of squares programming solver." *Proc. IEEE Conf. on Decision and Control*.

Prajna, S., A. Papachristodoulou, and F. Wu (2004): "Nonlinear control synthesis by sum of squares optimization: A Lyapunov-based approach." *Proceedings of the Asian Control Conference (ASCC), Melbourne, Australia*.

Prestel, A. and C. N. Delzell (2001): *Positive polynomials*. Springer-Verlag.

Putinar, M. (1993): "Positive polynomials on compact semi-algebraic sets." *Indiana Univ. Math. J.*, **42:3**, pp. 969–984.

Rantzer, A. (2006): "Relaxed dynamic programming in switching systems." *IEE Proceedings - Control Theory & Applications*, **153:5**, pp. 567 – 574.

Shamma, J. S. and D. Xiong (1997): "Linear nonquadratic optimal control." *IEEE Transactions on Automatic Control*, **42:6**, pp. 875–879.

Shor, N. Z. (1987): "Class of global minimum bounds of polynomial functions." *Cybernetics*, **23:6**, pp. 731–734.

Sturm, J. F. (1999): "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones." *Optimization Methods and Software*, **11–12**, pp. 625–653. Special issue on Interior Point Methods (CD supplement with software).

Wernrud, A. and A. Rantzer (2005): "On approximate policy iteration for continuous-time systems." In *The 44th IEEE Conference on Decision and Control and European Control Conference ECC.* Seville, Spain.

Wernrud, A. (2006): "Computations of approximate value functions for constrained control problems." In *The 17th International Symposium on Mathematical Theory of Networks and Systems*. Kyoto,Japan.

Wernrud, A. (2007): "Strategies for computing switching feedback controllers." In *American Control Conference*. New York City, USA.

Wernrud, A. (2008):"On constrained optimal control of linear systems with piecewise linear step cost." Submitted.

Youla, D. C., H. A. Jabr, and J. J. Bongiorno (1976a): "Modern Wiener-Hopf design of optimal controllers. part 1: The single-input case." *IEEE Trans. on Automatic Control*, **21**, June.

Youla, D. C., H. A. Jabr, and J. J. Bongiorno (1976b): "Modern Wiener-Hopf design of optimal controllers. part 2: The multivariable case." *IEEE Trans. on Automatic Control*, **21**, June.