Computational Methods for Optimal Control of Hybrid Systems

Hedlund, Sven

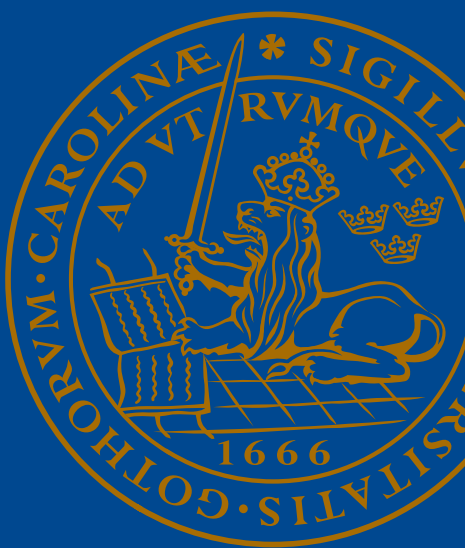
2003

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

Computational Methods for Optimal Control of Hybrid Systems

Sven Hedlund

Automatic Control



Computational Methods for Optimal Control of Hybrid Systems

Computational Methods for Optimal Control of Hybrid Systems

Sven Hedlund

Department of Automatic Control
Lund Institute of Technology
Lund, May 2003

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

ISSN 0280–5316
ISRN LUTFD2/TFRT--1068--SE

Printed in Sweden by Bloms i Lund Tryckeri AB.
Lund 2003

Abstract

This thesis aims to find algorithms for optimal control of hybrid systems and explore them in sufficient detail to be able to implement the ideas in computational tools. By hybrid systems is meant systems with interacting continuous and discrete dynamics. Code for computations has been developed in parallel to the theory.

The optimal control methods studied in this thesis are global, i.e. the entire state space is considered simultaneously rather than searching for locally optimal trajectories. The optimal value function that maps each state of the state space onto the minimal cost for trajectories starting in that state is central for global methods. It is often difficult to compute the optimal value function of an optimal control problem, even for a purely continuous system. This thesis shows that a lower bound of the value function of a hybrid optimal control problem can be found via convex optimization in a linear program. Moreover, a dual of this optimization problem, parameterized in the control law, has been formulated via general ideas from duality in transportation problems. It is shown that the lower bound of the value function is tight for continuous systems and that there is no gap between the dual optimization problems.

Two computational tools are presented. One is built on theory for piecewise affine systems. Various analysis and synthesis problems for this kind of systems are via piecewise quadratic Lyapunov-like functions cast into linear matrix inequalities. The second tool can be used for value function computation, control law extraction, and simulation of hybrid systems. This tool parameterizes the value function in its values in a uniform grid of points in the state space, and the optimization problem is formulated as a linear program. The usage of this tool is illustrated in a case study.

Contents

Acknowledgments

This tiny space is dedicated to colleagues and friends. I am very fortunate to have you all around, you are great! I could easily fill chapters describing how much you mean to me, but the main purpose of the book currently in your hands is to present my research.

First of all, I want to express my sincere gratitude to my supervisor Anders Rantzer. He is full of good ideas and is a true source of inspiration, in research as well as humanity. I also want to thank Karl-Erik Årzén for his guidance during my first years at the department.

Andrey Ghulchak, who worked at our department during several years of my graduate studies, was and still is a great asset, always willing to answer questions about mathematics. I have also thoroughly enjoyed the company of Magnus Gäfvert. Sharing office with him for several years has lead to lots of stimulating projects and discussions, not only related to automatic control.

I want to thank Leif and Anders for maintaining a sound computer system at our department as well as answering all my questions about computer programming, Rolf for building and supporting equipment necessary to put ideas into practice, and the cheerful secretaries Eva, Britt-Marie, and Agneta for taking care of everything.

Having got the fortunate opportunity to study in various countries, I want to thank all of those who worked on their doctoral theses at Electrical Engineering, University of Newcastle, Australia during my master thesis — you made me first interested in further academic research. I also want to thank Prof. Tetsuya Iwasaki who received me at Tokyo Institute of Technology, Japan, and Prof. Jan van Schuppen, who invited me to CWI, The Netherlands for your hospitality. It was a great pleasure to work with both of you.

I also want to thank my family and friends for filling my life with joy outside working hours. Thank you Agneta and Gunnar, Per and Marie, Anders and Karin, and all friends in Lund for your support.

My final words of gratitude are for my dear Monika, what would I do without you?

Sven

Financial Support

I have received financial support from various sources during my years as a PhD student. Support from the Swedish Research Council, European Project EP21911 “FAMIMO”, Esprit Project 28104 (H2C), and IST Project IST-2001-33520 (CC) is gratefully acknowledged. I would also like to thank The Scandinavia-Japan Sasakawa Foundation and “Ernhold Lundströms stiftelse” for travel grants.

1

Introduction

This thesis consists of a collection of papers related to optimal control of hybrid systems and a few additional chapters that further illustrates some benefits of the papers.

Section 1.1 of this introductory chapter briefly describes the contribution of each paper and subsequent chapters. Section 1.2 discusses various ways of modeling a hybrid system and puts the model used for this thesis into perspective. Section 1.3 presents current efforts in optimal control of hybrid systems.

1.1 Contribution of This Thesis

In optimal control, every possible trajectory (control signal/state trajectory combination) is assigned a positive number, called the cost of the trajectory. This way, it is possible to state the optimal control problem as to find the cheapest trajectory between two points in the state space.

In global optimization methods, i.e. methods in which the optimization considers the entire state space rather than searches for local minima, the *value function* plays an important role. To each control feedback law, there is a corresponding value function or cost-to-go function that maps every state of the state space onto the cost of the trajectory starting in that state. The *optimal* value function is the value function corresponding to the optimal control law.

Much of the work in this thesis relates to hybrid systems, i.e. systems that contain continuous time-driven dynamics that interacts with discrete event-driven dynamics. Most of the papers included are related to optimal value functions of optimal control problems for hybrid systems.

Paper A. Convex Dynamic Programming for Hybrid Systems

This paper shows how the optimal value function corresponding to a hybrid optimal control problem can be computed via a linear program (LP).

Any function that satisfies the constraints of the presented LP is a lower bound of the optimal value function.

The paper, that is an extension of [Hedlund and Rantzer, 1999], also presents a possible method of discretizing the LP to find lower bounds of optimal value functions numerically. An example shows that this discretization scheme can be used for hybrid systems with a complexity level of at least three continuous states.

Paper B. Duality Between Cost and Density in Optimal Control

Purely continuous systems are treated in this paper. An optimization problem that is dual to the continuous systems version of the optimal value function LP in Paper A is derived. The dual is used to prove that the lower bound of the optimal value function derived in Paper A is tight. The dual optimization problem is parameterized in the control law and an interesting property is that any (non optimal) control law that satisfies the inequalities of the dual problem is stabilizing.

Though standing on its own, in the context of hybrid systems this paper can be regarded as preparatory work for Paper C.

Paper C. Duality in Hybrid Optimal Control

This paper is an extension of Paper B to hybrid systems. The main result, the dual of the optimization problem stated in Paper A, is formulated for hybrid systems.

The paper tries to convey an intuitive understanding of the two different optimization approaches via a thorough discussion around simple examples.

Paper D. Hybrid Control Laws From Convex Dynamic Programming

This paper describes an attempt to discretize the dual optimization problem for hybrid systems.

Paper E. A Toolbox for Computational Analysis of Piecewise Linear Systems

This paper reports the development of a MATLAB toolbox for analysis and synthesis of piecewise affine systems (called piecewise linear systems in this paper). The toolbox is based on the theory presented in [Johansson, 2002].

The paper gives insight into the features and functionality of the toolbox via command listings and computational examples.

2. A Matlab Tool for Dynamic Programming for Hybrid Systems

This chapter presents a set of MATLAB commands, called CDP Tool, that makes the theory of Paper A easy accessible. CDP Tool contains com-

mands for finding approximations to the optimal value function, control law computation, and simulation. Some equations are presented as they are needed to understand the tool, though the main focus of this presentation is usage of the tool rather than implementation issues.

For a complete description of the commands of the tool, see the reference manual included in [Hedlund, 1999].

3. Case Study: CDP Tool in the Control Laboratory

This chapter documents a case for which the computational tool reported in Chapter 2 has been used successfully in practice. The case study aims to give insight into considerations that the CDP Tool user faces and to hint a general work flow working with the tool.

1.2 Hybrid System Models

A hybrid system consists of continuous dynamics that interacts with discrete dynamics. By continuous dynamics, we mean dynamics described by variables that take values from a continuous set. Similarly, discrete dynamics is described by a variable that takes values from a discrete set. A hybrid system usually also contains both time-driven and event-driven dynamics. Time driven dynamics means that the evolution of the corresponding variable is synchronized by a clock. Event-driven dynamics means that the changes in the corresponding variable are triggered by events, not necessarily at regular intervals, not necessarily known in advance. The continuous dynamics is often time-driven, while the discrete dynamics is event-driven.

The hybrid model used in this thesis consists of continuous and discrete states as well as continuous and discrete inputs.

As in all mathematical modeling, the choice of approach for hybrid modeling heavily depends on the purpose of the model. In computer science, hybrid modeling often means extending event-driven models to allow time-driven dynamics between the event occurrences, or to determine event occurrence times. In automatic control, it is common to extend time-driven models to allow events that drastically change the dynamics or trigger jumps in the continuous states. Hybrid systems are also suitable for modeling of systems with multiple time scales, where the fast dynamics can be treated as discrete, abrupt changes compared to the slower dynamics.

Below are some examples of hybrid systems to show what we have in mind. A more precise formulation of the hybrid system model used in this thesis will be given in the end of Section 1.2.

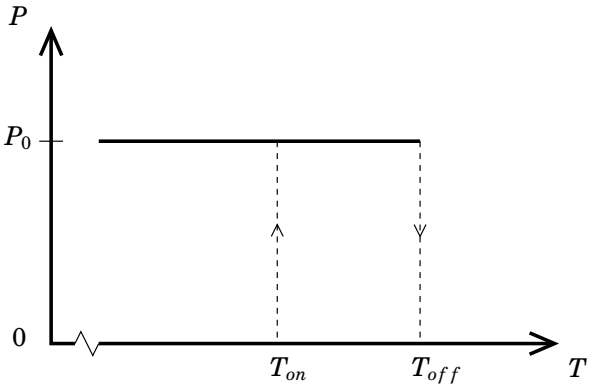


Figure 1.1 Thermostat power profile that shows the power fed to the radiator, P, as a function of the room temperature, T. The dashed lines indicate that the radiator is switched on when the temperature falls below T_{on} and switched off when the temperature rises above T_{off}.

Introductory Examples

EXAMPLE 1.1—THERMOSTAT CONTROLLED HEATING

Consider the problem of heating a room with a radiator. The radiator is controlled by a thermostat that turns the radiator on when the room temperature, T, falls below a threshold, T_{on}, and turns the radiator off when T rises above T_{off} ($T_{on} < T_{off}$). This hysteresis behavior is illustrated in Fig. 1.1 where T is the room temperature and P is the power that the radiator feeds the room.

This closed loop heating system can be interpreted as a hybrid system and one way of modeling the system is as a hybrid automaton according to Fig. 1.2.

The system has two discrete modes (represented by the ellipses), “no heating” and “heating”, and one continuous variable, T. Each discrete mode has an associated differential equation that governs the evolution of the continuous state; in the "no heating" mode the temperature tends towards zero, in the heating mode the temperature tends towards a stationary value T_{stat} ($> T_{off}$).

Each discrete mode also has a constraint, a *mode invariant* (written below the differential equation in Fig. 1.2), that must be satisfied in that mode. There are also arrows between the discrete modes that indicate possible mode switching. Along with each mode switch there is a constraint, a *guard* (written next to the base of the arrow). No mode switch can take place unless the corresponding guard is satisfied. □

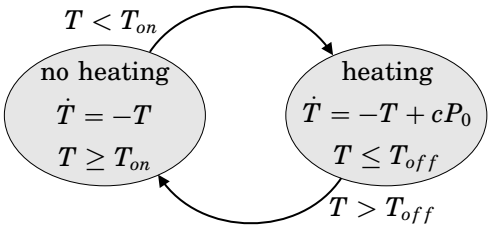


Figure 1.2 A hybrid automaton modeling the heating of a room. The "heating" mode is entered (the radiator is turned on) when the the temperature T falls below T_{on}. The system returns to "no heating" when T rises above T_{off}.

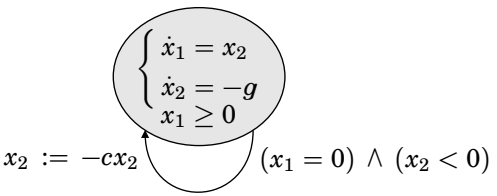


Figure 1.3 A hybrid automaton modeling a bouncing ball, where x_1 is the height of the ball above floor and x_2 is the velocity. There is an abrupt change of velocity $(x_2 := -cx_2)$ when the ball bounces.

EXAMPLE 1.2—BOUNCING BALL EXAMPLE

Consider the problem of a bouncing ball. Let the continuous state variable x_1 be the distance between the ball and the floor $(x_1 \geq 0)$. Let x_2 be the velocity of the ball. This example does not contain any discrete *state*, but it is convenient to model the bouncing of the ball as a discrete *event* in a hybrid system. Assume that each bounce can be modeled as an inelastic collision, where the ball looses some momentum. Each bounce will then result in a *jump* in the velocity state such that $x_2^+ = -cx_2^-$, where x_2^- and x_2^+ means the velocity just before the bounce and just after the bounce respectively, and c is a constant $0 < c < 1$.

The entire trajectory can be modeled with a hybrid automaton as in Fig. 1.3. In the figure, the state jump is indicated next to the tip of the arrow using the assignment operator ":=". The constant g is the acceleration of gravity, and the symbol "$\wedge$" denotes the logical operator "and".

□

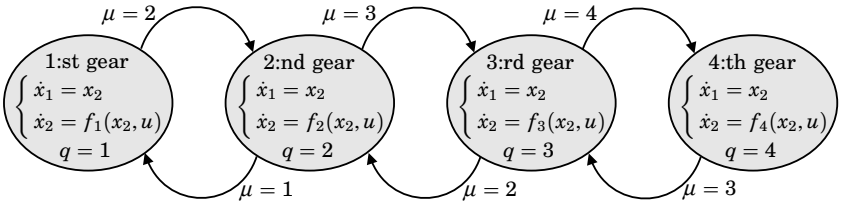


Figure 1.4 A model of a car with a manual gear box, where x_1 is the longitudinal position and x_2 is the velocity. There is a continuous input u corresponding to the throttle, and a discrete input μ corresponding to the gear.

EXAMPLE 1.3—A CAR WITH A MANUAL GEAR BOX

The former examples of hybrid systems have been autonomous. This example has both a continuous input and a discrete input. A car that travels along a predefined path can be modelled with two continuous states, the position, x_1, and the velocity, x_2. It also has a discrete state, the gear, q. The gas pedal is the continuous input, denoted u, and the gear lever is the discrete input, denoted μ. A schematic view of a system with four gears is shown in Fig. 1.4

As seen from the figure, there are different acceleration functions f_i for different gears i, $(1 \leq i \leq 4)$. Since all gears have their maximum efficiency at different speeds, the f_i:s will have their peaks for different values of x_2. The figure also shows that certain combinations of states and inputs are not allowed, i.e. the *domain* of the model is limited. The model does for instance not accept switching from gear $= 1$ to gear $= 4$. □

The list of applications that can be modelled as hybrid systems is enormous. Additional examples include constrained robotic systems, automated highway systems, air traffic management systems, electric circuits with diodes and switches, disk drives, and computer controlled systems, see [IEEE, 1998; IEEE, 2000; van der Schaft and Schumacher, 2000] and the rest of this thesis.

Modeling Pitfalls

Modeling hybrid systems, there is a risk of producing models that are unreasonable, either physically or mathematically. Modeling issues such as existence and uniqueness of trajectories certainly have their counterparts also in pure continuous systems, but the possibly higher complexity of hybrid systems often make these problems more difficult to discover in modeling. It should also be mentioned, though, that hybrid systems are often intentionally modelled without uniqueness to incorporate uncertainty.

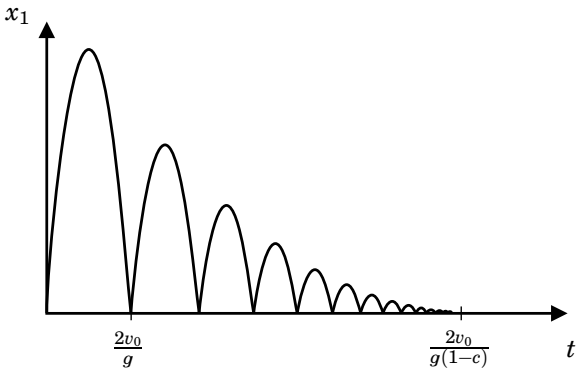


Figure 1.5 Simulation of the bouncing ball automaton where x_1 is the distance between the ball and the floor and t is the time.

One issue that is unique for hybrid systems is the possibility of an infinite number of mode switches in finite time, called *Zeno* behavior. One can argue that physical systems do not exhibit Zeno behavior. Having modelled a hybrid system, however, and having created a model that seems reasonable from a physical point of view, a refined analysis of the model may show unintentionally introduced Zeno behavior. Fig. 1.5 shows a trajectory of the bouncing ball automaton. The states are initialized to $(x_1, x_2)(0) = (0, v_0)$, where $v_0 > 0$ is the initial upwards velocity. As indicated in the figure, there is an infinite number of bounces up to $t = \frac{2v_0}{g(1-c)}$ and the simulation reaches no further. A special case of Zenoness is livelock, an infinite number of switches at the same time. An example of this is chattering which may appear in the discontinuous dynamics of sliding mode control.

Various Formal Models

Hybrid systems is an immature research area, and there have been various efforts to state an overall unified model[Branicky *et al.*, 1994; Bensoussan and Menaldi, 1997], as well as finding common ground between different models[Heemels *et al.*, 2001]. Generally, a hybrid system can be described as a hybrid automaton, $\mathcal{H} = (X, \bar{Q}, \Omega_u, \Omega_\mu, f, \text{Inv}, G, R)$ where

- $X \subset \mathbf{R}^n$ is the state space for the continuous variable.
- $\bar{Q} = \{1, 2, \ldots, Q\}$ is the mode set, the state space of the discrete variable.
- $\Omega_u \subset \mathbf{R}^p$ is a set of inputs (disturbances as well as controlled inputs).

- Ω_μ is a finite set of discrete inputs (disturbances as well as controlled inputs).
- f is a map from $X \times \bar{Q} \times \Omega_u$ to $\mathbf{R}^n$ that describes the continuous, time-driven evolution, as explained below.
- the mode invariant $\text{Inv} \subset X \times \bar{Q} \times \Omega_u \times \Omega_\mu$, the guard $G \subset X \times \bar{Q} \times \Omega_u \times \Omega_\mu$, and the reset map $R : X \times \bar{Q} \times \Omega_u \times \Omega_\mu \to 2^{X \times \bar{Q}}$ describes the discrete, event-driven evolution[1]. The invariant tells where continuous evolution is possible, the guard tells where discrete transitions are possible, and the reset map gives possible new states resulting from a transition.

The pair $(x, q) \in X \times \bar{Q}$, consisting of the continuous state x and the discrete state q, is called the state of H. The values of the discrete state are called different discrete modes. The possible evolutions of the automaton can be described as follows. Starting at an initial state, $(x_0, q_0) \in X \times \bar{Q}$, given the inputs $u \in \Omega_u$ and $\mu \in \Omega_\mu$, the continuous state x evolves according to $\dot{x}(t) = f(x(t), q(t), u(t))$ while the discrete state $q(t)$ remains constant. Continuous evolution can go on as long as $(x, q, u, \mu) \in \text{Inv}$. If the situation arises during the evolution that $(x, q, u, \mu) \in G$, a transition is possible to a new state. The set of possible new states is given by $R(x, q, u, \mu)$. The transition from the old state (x, q) to the new state (x', q') is called a jump if $x' \neq x$ and a switch if $q' \neq q$. After the transition, there are either new transitions or the continuous evolution resumes and the whole process is repeated.

The above general model incorporates uncertainty in different ways. Necessary (but not sufficient) additional constraints for uniqueness of trajectories include $\text{Inv} \bigcap G = \emptyset$ and $R(x, q, u, \mu)$ being a singleton for each $(x, q, u, \mu) \in X \times \bar{Q} \times \Omega_u \times \Omega_\mu$. The issue of determinism of hybrid systems is outside the scope of this thesis, but is treated in for example [Lygeros *et al.*, 1999].

EXAMPLE 1.4—BOUNCING BALL, FORMAL MODEL

Consider again the bouncing ball of Ex. 1.2. Following the above notation, $X = \{(x_1, x_2) \in \mathbf{R}^2 | \, x_1 \geq 0\}$, $\bar{Q} = \{1\}$. Since the system is autonomous, the sets Ω_u and Ω_μ are empty and can be omitted. The continuous dynamics is

$$f(x, 1) = \begin{bmatrix} x_2 \\ -g \end{bmatrix}$$

and the discrete dynamics is given by $G = \{(x_1, x_2, q) \in X \times \bar{Q} \, | \, x_1 =$

[1]The expression $2^{X \times \bar{Q}}$ denotes the power set (the set of all subsets of) $X \times \bar{Q}$

$0,\ x_2 < 0,\ q = 1\}$, $\mathrm{Inv} = (X \times \bar{Q}) \setminus G$,

$$R(x,1) = \begin{cases} \{(x_1, -cx_2, 1)\} & \text{if } (x,1) \in G \\ \varnothing & \text{otherwise} \end{cases}$$

□

EXAMPLE 1.5—FORMAL MODEL OF A CAR WITH A MANUAL GEAR BOX,
Consider again the car in Ex. 1.3. For this example, $X \subset \mathbf{R}^2$ is a hyperrectangle reflecting the positions under consideration and the velocity constraints of the vehicle. The number of discrete modes $Q = 4$, i.e. $\bar{Q} = \{1,2,3,4\}$, $\Omega_u = \{u \in \mathbf{R} \,|\, 0 \le u \le u_{\max}\}$ for some positive $u_{\max}$, and $\Omega_\mu = \{1,2,3,4\}$.

$$f(x,q,u) = \begin{bmatrix} x_2 \\ f_q(x_2,u) \end{bmatrix}$$

where $f_q(x_2,u)$, $q \in \bar{Q}$ are the functions in Fig. 1.4. The sets governing the event-based evolution are $\mathrm{Inv} = \bigcup_{i=1}^{Q} X \times \{i\} \times \Omega_u \times \{i\}$, $G = \bigcup_{(q,\mu)\in G_{q\mu}} X \times \{q\} \times \Omega_u \times \{\mu\}$ where $G_{q\mu} = \{(1,2),\ (2,1),\ (2,3),\ (3,2),\ (3,4),\ (4,1)\} \subset \bar{Q} \times \Omega_\mu$, and, with slight abuse of notation,

$$R(x,q,u,\mu) = \begin{cases} \{(x,\mu)\} & \text{if } (x,q,u,\mu) \in G \\ \varnothing & \text{otherwise} \end{cases}$$

□

Hybrid automata models have been used in slightly different forms by many researchers in both control and computer science, for example [Göllü and Varaiya, 1989; Stiver and Antsaklis, 1992; Branicky *et al.*, 1994; Alur *et al.*, 1997; van der Schaft and Schumacher, 1998; Sussmann, 1999; Tomlin *et al.*, 2000; Johansson *et al.*, 2003]. Some of the models include an output as a function of states and inputs. Because of the nature of the control problem studied in this thesis, we are not interested in an output and will not carry with us this extra notational burden for each model. Some of the models are specified with a set of possible initial states.

Switched Systems *Switched systems* is a class of hybrid systems consisting of a family of time driven (continuous time or discrete time) subsystems and a possibility to switch between them. There exist various system definitions that offer different switching possibilities. One approach is to regard the switching as an external event, an input that can or cannot be controlled, in which case there is no need for a discrete state. Another

approach is to regard the switching as internally generated (the switching dynamics fits into Inv, G, and R above), in which case there is no need for a discrete input. Switched systems are treated in for example [Hou *et al.*, 1996; Branicky, 1998; Xu and Antsaklis, 2000; Hespanha *et al.*, 2002]. Switched systems do not feature jumps in the continuous state.

An important special case of the latter approach are the piecewise affine models.

Piecewise Affine Models The reason for the popularity of piecewise affine (PWA) models is that they make a good compromise between generality and ease of analysis. They are one of the simplest extensions of linear systems that can model a variety of nonlinear systems arbitrarily well.

For PWA systems, the continuous part of the state space is partitioned into convex polyhedra[2] $X_1, X_2, \ldots, X_Q$ such that $X = \bigcup_{i=1}^{Q} X_i$, $X_i \bigcap X_j = \emptyset$ for $i \neq j$. The time driven dynamics is affine in the states and inputs, i.e.

$$\dot{x}(t) = f\big(x(t), q(t), u(t)\big) = A_q x(t) + B_q u(t) + c_q \qquad \text{for } x \in X_q$$

where $A_q \in \mathbf{R}^{n\times n}$, $B_q \in \mathbf{R}^{n\times p}$, $c_q \in \mathbf{R}^{n\times 1}$. A discrete time counterpart is also commonly used:

$$x(k+1) = A_q x(k) + B_q u(k) + c_q \qquad \text{for } x \in X_q$$

Note that the discrete state q is redundant in PWA models, i.e. the above model (continuous time version) is equivalent to

$$\dot{x}(t) = \hat{f}\big(x(t), u(t)\big)$$

with the function $\hat{f}(x,u) = f(x,q,u)$ for $q \in X_q$ possibly discontinuous in x on the boundary of the partitioning polyhedra. It is sometimes beneficial, however, to view the abrupt changes in the dynamics as hybrid mode switches.

In their most general form, PWA systems are not only partitioned in the state space, but also in the input space, i.e. $X_q \times \Omega_{u,q} \subset X \times \Omega_u$.

For information about discrete time PWA systems, see for example [Sontag, 1981; Heemels *et al.*, 2001; Bemporad and Morari, 1999a] For continuous time PWA systems see for example [Johansson, 2002; Heemels, 1999].

[2] A convex polyhedron is a set that can be described as intersections of a finite number of half spaces.

Timed Automata In traditional model checking, a high-level description of a discrete event system is compared against a logical correctness requirement to discover inconsistencies. In real time systems, where the correctness does not only depend on the results of the computations but also on the time when they are delivered, there is a need for explicit modeling of time. The timed automaton, which is a discrete event system extended with continuous timer states, was introduced to take care of this problem[Alur and Dill, 1994].

The continuous time-driven dynamics of timed automata is simple, $\dot{x}_i = 1$, $i = 1, 2, \ldots, N$, with possible jumps in the continuous states (resetting of the timers). Though the timed automaton is rudimentary for a hybrid system, its applicational relevance has lead to extensive research lately. There exist generalizations such as multirate timed automata where $\dot{x}_i = c_i$, $c_i \in \mathbf{R}$, and rectangular timed automata where $\dot{x}_i \in [\underline{c}_i, \overline{c}_i]$, $\underline{c}_i \leq \overline{c}_i \in \mathbf{R}$[Henzinger *et al.*, 1998].

Manufacturing Model The combination of time-driven and event-driven dynamics is a natural framework for manufacturing processes. The physical characteristics of production parts undergo changes at various operations described by time-driven models, while the timing control of operations is described by event-driven models[Pepyne and Cassandras, 2000].

Represented in the general hybrid automaton model, the manufacturing system easily grows to a high dimension. In its most general form, each manufactured part has its continuous state, for example temperature or geometry, and each possible configuration of different parts at different machines requires its own discrete mode. There is no discrete input to this system, but the discrete state is affected by the continuous input (for example higher power to the furnace means less heating time and earlier switching to another part).

This rather special case of hybrid system is listed here for its practical interest and because its special structure makes it possible to analyze and control despite its complexity.

The Hybrid System Model Used in This Thesis The model presented below has been used (with slight variations) in most of the papers included in this thesis. It has much in common with the hybrid automaton $\mathcal{H}$ and some differences will be commented after the following definition.

Let $X \subset \mathbf{R}^n$ and $\Omega_u \subset \mathbf{R}^p$ denote the continuous state space and input respectively. Let the finite sets $\bar{Q}$ and Ω_μ denote the discrete state space and input respectively. The time driven dynamics is governed by the function $f : X \times \bar{Q} \times \Omega_u \to \mathbf{R}^n$, and the event driven dynamics is given by $\nu : X \times \bar{Q} \times \Omega_\mu \to \bar{Q}$.

The hybrid system model is then defined via the trajectories it accepts. Given a time interval $[t_i, t_f] \subset \mathbf{R}$, the function $u : [t_i, t_f] \to \Omega_u$, and the piecewise constant function $\mu : [t_i, t_f] \to \Omega_\mu$, a trajectory of the hybrid system

$$\begin{cases} \dot{x}(t) = f(x(t), q(t), u(t)) \\ q(t) = \nu(x(t), q(t^-), \mu(t)) \end{cases} \tag{1.1}$$

is defined as the collection (T, x, q) with the following properties. $T = t_0, t_1, \ldots, t_M$ is an increasing sequence of $M+1$ real numbers with $t_0 = t_i$ and $t_M = t_f$, $x : [t_0, t_M] \to X$ is absolutely continuous, $q : [t_0, t_M] \to \bar{Q}$ is constant in each interval $[t_k, t_{k+1})$, $k = 0, 1, \ldots, M-1$, and

$$\begin{cases} \dot{x}(t) = f(x(t), q(t), u(t)) & \text{for almost all } t \in [t_0, t_M] \\ q(t) = \nu(x(t), q(t), \mu(t)) = q(t_k) & t \in (t_k, t_{k+1}) \\ q(t_{k+1}) = \nu(x(t_{k+1}), q(t_k), \mu(t_k)) & k = 0, 1, \ldots, M-1 \end{cases} \tag{1.2}$$

This model differs from the hybrid automaton, $\mathcal{H}$, in that the event-driven dynamics is described by the ν-function instead of the triple Inv, G, and R. This model does allow a finite number of switches (autonomous as well as controlled), but no jumps. Thus it can be used for Ex. 1.1 and Ex. 1.3 but not Ex. 1.2. The model does *not* allow the kinds of uncertainty that the automaton $\mathcal{H}$ incorporates by $\text{Inv} \bigcap G \neq \emptyset$ or R not being a singleton.

EXAMPLE 1.6—THERMOSTAT CONTROLLED HEATING REVISITED
The continuous variable of Ex. 1.1 is T. Let $q \in \bar{Q} = \{1, 2\}$ be the discrete variable, where $q = 1$ corresponds to the radiator being switched off and $q = 2$ corresponds to the radiator being active. Since the closed loop system is autonomous, it can be described as

$$\begin{cases} \dot{T}(t) = f(T(t), q(t)) \\ q(t) = \nu(T(t), q(t^-)) \end{cases}$$

where $f(T, 1) = -T$, $f(T, 2) = -T + cP_0$, and

$$\nu(T, q) = \begin{cases} 1, & \text{if } (q = 1 \wedge T \geq T_{on}) \vee (q = 2 \wedge T > T_{off}) \\ 2, & \text{if } (q = 1 \wedge T < T_{on}) \vee (q = 2 \wedge T \leq T_{off}) \end{cases}$$

□

1.3 Optimal Control of Hybrid Systems

The purpose of optimal control is to give a general systematic method to synthesize a control law with certain properties. These properties are specified in an optimization criterion, or a cost function. This is a mathematical expression in the state variables (and possibly the control signals) that should be minimized subject to the process constraints. For hybrid systems, the cost function in general involves an integral cost accumulated along continuous evolution and switching costs associated with discrete transitions. The cost function can be used to penalize various quantities such as energy consumption, deviation from a desired set point, areas of the state space that are not considered safe etc.

Once a cost function for the problem is specified, the control synthesis is transformed to an optimization problem. In this way, it would ideally be possible to formally state once and for all what is good control for a certain process, and then apply suitable mathematical tools to find the controller that best meets the specifications.

One problem, however, is that it is often difficult to find a good cost function a priori. It is therefore common practice to choose a cost function, compute the controller, run experiments (by simulation or controlling the real process), evaluate the results, go back to the first step of adjusting the cost function. Thus, the design procedure will still have to be iterative.

Another potential obstacle is that the resulting mathematical problem may be hard to solve. It often involves non-convex optimization of a nonlinear function subject to dynamic constraints. This obstacle may be particularly visible in optimal control of hybrid systems, where the constraints of combined time-driven and event-driven dynamics in general are very complex. In practical systems, it is thus often desirable to reduce complexity by optimizing a selection of subcomponents of the process to make the overall result "almost optimal." Various approximations are also common, modifying the original specification of the problem, e.g by linearization of nonlinear dynamics or by discretizing the continuous state space. Below are various methods for optimal control of hybrid systems.

Variational Formulation

The maximum principle was formulated for continuous systems in 1962 by Pontryagin [Pontryagin *et al.*, 1962]. Given an initial state, it is used to find an optimal trajectory by calculus of variation.

Consider the simple example of finding the minimum of the differentiable scalar function $g : \mathbf{R}^n \to \mathbf{R}$. It is well known that a necessary constraint for x^* to be a minimum is that $(dg/dx)(x^*) = 0$. If a minimum exists, it can thus be located by finding all points that satisfy the derivative constraint and compare the value of g for each point. The maximum

principle gives similar necessary constraints on a trajectory to minimize a cost function in a dynamical system. Each trajectory that satisfies the constraints of the maximum principle is a candidate for the optimal trajectory. Having found a (possibly infinite) number of candidates, the optimal one has to be selected by other means. The candidates can sometimes be found analytically (such as in linear quadratic control), but often have to be computed by numerical methods.

Being a valuable method for optimal control of continuous systems, there have been recent efforts to extend the maximum principle to hybrid systems[Sussmann, 1999; Grammel, 1999; Riedinger *et al.*, 1999]. However, the local optimization of the maximum principle relies on comparisons between neighboring trajectories — comparisons which have a different nature in a system with mode switches. The maximum principle can still be used on the continuous evolution of the trajectory between switches.

Value Function Based Approaches

Dynamic Programming The term dynamic programming was introduced by Bellman in 1957 [Bellman, 1957]. The basic idea is the principle of optimality, which says that in any state along an optimal trajectory, the remaining part must constitute an optimal trajectory when that state is considered as an initial state.

The *value function* or the *cost-to-go function* is central in dynamic programming. This is a function that maps every state onto the cost for a trajectory starting in that state. The principle of optimality can be translated to mathematics in terms of constraints on the optimal value function and the corresponding control signal. The constraint equation, called the Hamilton-Jacobi-Bellman (HJB) equation, is a partial differential equation for the continuous systems case and a difference equation for the discrete systems case.

One of the difficulties in applying the HJB equation for systems with continuous dynamics is that the value functions of many optimal control problems are not differentiable. The HJB equation still make sense, however, if non-classical interpretation of solutions to differential equations are used, such as viscosity solutions[Bardi and Capuzzo-Dolcetta, 1997; Bensoussan and Menaldi, 1997].

Being a global optimization method in that the value function needs to be computed for the whole state space rather than along a trajectory, the complexity of all existing numerical methods grows exponentially with the dimension of the state space. Bellman referred to this obstacle as “the curse of dimensionality.” Nevertheless, it is possible to derive an analytic solution for the simple but important case of quadratic loss function and

linear dynamics.

Lately, versions of the Hamilton-Jacobi-Bellman equation for hybrid systems have been formulated[Bensoussan and Menaldi, 1997; Branicky and Mitter, 1995].

Bounds on the Value Function As shown in this thesis, see for example Paper B, it is possible to replace the HJB equation by an *inequality* constraint to avoid dealing with a possibly non-differentiable optimal value function. The resulting linear inequality ensures that the constrained function is a lower bound of the optimal value function.

A similar approach for optimal control of piecewise affine systems using a piecewise quadratic cost function was used in [Johansson, 2002], where the resulting optimization problem is a linear matrix inequality.

Model Predictive Control

In Model Predictive Control (also called receding horizon control), a performance index is minimized over a finite horizon, subject to the constraints of the dynamical system. For each measurement, a sequence of control signals is computed based on a prediction of the future evolution of the system. This sequence is applied to the process until new measurements are available and the procedure is repeated by computing a new control signal sequence. The control signals are computed to optimize tracking performance while avoiding possible constraint violations. Model predictive control has been successfully applied to discrete time piecewise affine systems, where each control sequence computation can be formulated as a mixed integer quadratic program[Bemporad and Morari, 1999a]. There are also other algorithms for MPC of piecewise affine systems [Schutter and van den Boom, 2001].

Other Methods

Various methods exist that utilize the structure of certain subclasses of hybrid systems. A common approach is to split the optimization algorithm in a discrete part and a continuous part rather than to optimize simultaneously over discrete and continuous variables. An outer loop of the algorithm takes care of the discrete events by clever choice of different number of switchings and order of discrete modes. The inner loop finds the optimal continuous evolution given the switching scheme dictated by the outer loop. Examples of this include optimal control of switched systems[Xu and Antsaklis, 2000] and optimal control in the manufacturing model[Pepyne and Cassandras, 2000].

Reachability

The reachability problem is a well known and extensively studied prob-

lem. It is the question of whether, under given system dynamics, a given set of states can be reached from a given set of initial conditions. To see that the reachability problem is equivalent to the optimal control problem, consider the following example. Given the dynamics of the purely continuous system

$$\dot{x} = f(x,u),\ x(0) = x_0 \tag{1.3}$$

and the constants $c > 0$ and $T > 0$, is $\min_u \int_0^T l(x,u)dt < c$? Another equivalent question is whether there exists a state (x,z) that is reachable at time T with $z(T) < c$ in a system where (1.3) has been extended with $\dot{z} = l(x,u)$, $z(0) = 0$. This way, optimal control problems can be cast as a reachability problems. Conversely reachability problems can be cast as optimal control problems. Below are some examples of reachability studies for hybrid systems.

Controller Synthesis as a Game When designing safety critical control systems, it is beneficial to view the evolution of trajectories as a game between the controller and disturbances. One can pose safety games to meet safety specifications[Tomlin *et al.*, 2000]. The controller is computed to prevent the trajectory from leaving a desired region of the hybrid state space, the safe set, under all possible disturbances. Control synthesis involves computations of reachable sets which are found via Hamilton-Jacobi equations. Applicational examples involve collision avoidance of various vehicles as platooning cars and aircrafts.

There is an ongoing development of computational tools for estimation of reachable sets of general hybrid systems, for example [Silva and Krogh., 2000; Asarin *et al.*, 2000], as well as of simpler abstractions such as generalizations of timed automata [Daws *et al.*, 1994; Larsen *et al.*, 1995; Henzinger *et al.*, 1995].

Paper A

Convex Dynamic Programming for Hybrid Systems

Sven Hedlund and Anders Rantzer

Abstract

A classical linear programming approach to optimization of flow or transportation in a discrete graph is extended to hybrid systems. The problem is finite-dimensional if the state space is discrete and finite, but becomes infinite-dimensional for a continuous or hybrid state space. It is shown how lower bounds on the optimal value function can be computed by gridding the continuous state space and restricting the linear program to a finite-dimensional subspace. Upper bounds can be obtained by evaluation of the corresponding control laws.

Keywords Hybrid systems, optimal control, dynamic programming, linear program

1. Introduction

For several decades, linear programming has been one of the main theoretical and computational tools for analysis and optimization of discrete systems. This includes problems of optimal transportation and optimal flow in a network [Koopmans, 1947; Ford and Fulkerson, 1962; Bertsekas and Tsitsiklis, 1996]. The objective of the current paper is to extend the computational linear programming approach to hybrid systems, i.e. systems that involve interaction between discrete and continuous dynamics.

Practical control systems typically involve switching between several different modes, depending on the range of operation. Even if the dynamics in each mode is simple and well understood, automatic mode switching can give rise to unexpected phenomena. Moreover, many phenomena can be described either by a discrete model or a continuous one, depending on the context and purpose of the model [Antsaklis and Nerode, 1998]. Consider for example an asynchronous discrete-event driven thermostat, which discretizes temperature information as {too cold, normal, too hot}.

Basic aspects of hybrid systems were treated in [Ezzine and Haddad, 1989] and [Utkin, 1977]. For stability analysis, see [Branicky, 1998] and references therein. The reformulation of a nonlinear optimal control problem in terms of infinite-dimensional linear programming has previously been used for continuous time systems in [Rantzer and Johansson, 2000] and is closely connected to ideas of [Vinter, 1993; Rachev and Rüschendorf, 1998].

It should be noted that there is a close connection between optimal control and reachability. A control system can be extended with an extra state that integrates a cost along the trajectories. Hence, a certain control cost is achievable if and only if the corresponding state in the extended system is reachable. Conversely, reachability of a certain state can be investigated by solving the optimal control problem to get there in minimum time. Verification (reachability analysis) of discrete event systems and timed automata is an extensively studied topic in computer science cf. [Henzinger *et al.*, 1998; Alur *et al.*, 1997; Bemporad and Morari, 1999b].

Lately, various efforts have been made to extend the classical optimal control methods to hybrid systems. Hybrid versions of the maximum principle have been presented in [Riedinger and Iung, 1999; Sussmann, 1999; Piccoli, 1999]. Dynamic programming for hybrid systems is discussed in [Branicky and Mitter, 1995; Bensoussan and Menaldi, 1997]. In this paper, it is shown how strict lower bounds on the optimal value function can be computed by gridding the continuous state space and restricting the linear program to a finite-dimensional subspace. Upper bounds can be obtained by evaluation of the corresponding control laws. Computational examples are given with up to three dimensions in the

continuous state space.

Below, the set of all integers will be denoted $\mathbf{Z}$. The set of strictly positive real numbers will be denoted $\mathbf{R}^+$.

2. Problem Formulation

DEFINITION 1
Let Q and Ω_μ be finite sets, while $X \subset \mathbf{R}^n$ and $\Omega_u \subset \mathbf{R}^m$. Let $\nu : X \times Q \times \Omega_\mu \to Q$ and for every $q \in Q$ let $f_q : X \times \Omega_u \to \mathbf{R}^n$. A *solution (trajectory)* of the hybrid system

$$\begin{cases} \dot{x}(t) = f_q(x,u) \\ q(t) = \nu(x(t), q(t^-), \mu(t)) \end{cases} \tag{1}$$

will be defined given $u : [t_0, t_M] \to \Omega_u$, a finite sequence of real numbers $t_0 < t_1 < t_2 < \ldots < t_M$, and $\mu : [t_0, t_M] \to \Omega_\mu$ constant in each interval $[t_k, t_{k+1})$.

The pair (x,q), where $x : [t_0, t_M] \to X$ is absolutely continuous and $q : [t_0, t_M] \to Q$ is constant in each interval $[t_k, t_{k+1})$, $k = 0, 1, \ldots, M-1$, is called a trajectory of the hybrid system (1) if

$$\begin{cases} \dot{x}(t) = f_{q(t)}(x(t), u(t)) & \text{for almost all } t \in [t_0, t_M] \\ q(t) = \nu(x(t), q(t), \mu(t)) = q(t_k) & t \in (t_k, t_{k+1}) \\ q(t_{k+1}) = \nu(x(t_{k+1}), q(t_k), \mu(t_k)) & k = 0, 1, \ldots, M-1 \end{cases} \tag{2}$$

□

Note that the second equation of (1) gives rise to *autonomous* switching in points (x,q) where $\nu(x,q,\mu) \neq q$, $\forall \mu \in \Omega_\mu$. The time argument, t, will often be omitted in the sequel.

The optimal control problem is to minimize the cost function

$$J(x_0, q_0, u(\cdot), \mu(\cdot), t_M, M) = \int_{t_0}^{t_M} l_{q(t)}(x(t), u(t)) dt + \sum_{k=1}^{M} s(x(t_k), q(t_{k-1}), q(t_k)) \tag{3}$$

with respect to $u(\cdot)$, $\mu(\cdot)$, t_M, and M subject to (2) given an initial state (x_0, q_0) at time t_0, and a fixed set of possible final states, $(x,q)(t_M) \in Y_M \subset X \times Q$.

The function $s(x,q,r) > \varepsilon > 0$ is a cost for switching from discrete state q to r, the continuous part being x just before the switch. Note that $s(\cdot) > \varepsilon > 0$ limits the number of jumps in solutions close to optimality.

The framework developed in this paper would also allow the number of continuous states to vary with the discrete mode according to $\dot{x}_q(t) = f_{q(t)}(x_q(t), u_q(t))$, where $x_q(t) \in X_q \subset \mathbf{R}^{n(q)}$, $u_q(t) \in \Omega_{u_q} \subset \mathbf{R}^{m(q)}$. The usage of the system description (1), however, will simplify notation.

Also the possibility of *state jumps* [Bensoussan and Menaldi, 1997; Branicky and Mitter, 1995] has been omitted to keep the notational complexity at a reasonable level.

3. Hybrid Dynamic Programming

PROPOSITION 1
Let $X = \cup_{k=1}^{N} X_k$ where $X_1, \ldots, X_N$ are closed polyhedra with disjoint interior, with Q, Ω_u, Ω_μ, f_q, and ν defined as in Definition 1. Let $s : X \times Q \times Q \to (0, \infty]$ and for $q \in Q$ let $l_q : X \times \Omega_u \to [0, \infty]$. Suppose that $V_k \in \mathcal{C}^1(X_k \times Q, \mathbf{R})$ with $V_k(x, q) = V_j(x, q)$ for $x \in X_k \cap X_j$, $q \in Q$. Let $Y_M \subset Y \subset X \times Q$ and $V(x, q) = V_k(x, q)$ for $x \in X_k$, $q \in Q$. If for almost all $(x, q) \in Y \backslash Y_M$

$$0 \leq \frac{\partial V}{\partial x}(x, q) f_q(x, u) + l_q(x, u) \qquad u \in \Omega_u \tag{4}$$

$$0 \leq V(x, \nu(x, q, \mu)) - V(x, q) + s(x, q, \nu(x, q, \mu)) \qquad \mu \in \Omega_\mu \tag{5}$$

$$0 \geq V(x_M, q_M) \qquad \forall (x_M, q_M) \in Y_M \tag{6}$$

then

$$\int_{t_0}^{t_M} l_{q(t)}(x(t), u(t))dt + \sum_{k=1}^{M} s(x(t_k), q(t_{k-1}), \nu(x(t_k), q(t_{k-1}), \mu(t_k))) \geq V(x(t_0), q(t_0))$$

for every solution to (1) that is contained in Y with $(x, q)(t_M) \in Y_M$. □

Proof. Let $\hat{u}(\cdot)$ and $\hat{\mu}(\cdot)$ be control signals that drive the system from the initial state $(x_0, q_0) \in Y$ at time t_0 to $(x_M, q_M) \in Y_M$ at time t_M. Let

$x_k = x(t_k)$ and $q_k = q(t)$, $t_k \le t < t_{k+1}$. Then

$$\begin{aligned}
&J(x_0, q_0, \hat{u}(\cdot), \hat{\mu}(\cdot), t_M, M) \\
&= \sum_{k=0}^{M-1} \int_{t_k}^{t_{k+1}} l_{q_k}(x, \hat{u}) dt + \sum_{k=1}^{M} s(x_k, q_{k-1}, \nu(x_k, q_{k-1}, \hat{\mu}(t_k))) \\
&\ge \sum_{k=0}^{M-1} \int_{t_k}^{t_{k+1}} -\frac{\partial V}{\partial x}(x, q_k) f_{q_k}(x, \hat{u}) dt + \sum_{k=1}^{M} \{V(x_k, q_{k-1}) - V(x_k, q_k)\} \\
&= \sum_{k=0}^{M-1} \{V(x_k, q_k) - V(x_{k+1}, q_k)\} + \sum_{k=1}^{M} \{V(x_k, q_{k-1}) - V(x_k, q_k)\} \\
&= V(x_0, q_0) - V(x_M, q_M) = V(x_0, q_0)
\end{aligned}$$

□

For the purely discrete case, the value function,

$$V^\star(x, q) \equiv \min_{u(\cdot), \mu(\cdot), t_M, M} J(x_0, q_0, u(\cdot), \mu(\cdot), t_M, M),$$

is obtained from the LP, i.e. $\sup V = V^\star$. Continuous dynamics adds difficulty, however, and the bound above may in general not be tight, i.e. $\sup V(x, q) \le V^\star(x, q)$.

For purely continuous systems, conditions for tightness have been derived in [Vinter, 1993]. The theory needed, however, is quite advanced and an extension to the hybrid case falls outside the scope of this paper.

4. Discretization

Utilizing a computer to solve (4)-(6) for a specific control problem, a straight forward approach is to grid the state space and require the inequalities to be met at a set of uniformly distributed points in Y. This approximation will, however, not guarantee a lower bound on the optimal cost, unless the nature of f_q and V *between* the grid points is taken into consideration.

For uniform gridding of $\mathbf{R}^2$, let

$$x_{jk} = jhe_1 + khe_2, \qquad j, k \in \mathbf{Z},\ h \in \mathbf{R}^+, \tag{7}$$

where e_1 and e_2 are unit vectors along the coordinate axes, and h is the

grid size. Also let

$$X^{jk} = \{x_{jk} + \theta_1 h e_1 + \theta_2 h e_2 : -1 \le \theta_i \le 1\} \tag{8}$$

$$(\underline{f}_q^{jk})_i = \min_{x \in X^{jk}, u \in \Omega_u} (f_q(x,u))_i \tag{9}$$

$$(\overline{f}_q^{jk})_i = \max_{x \in X^{jk}, u \in \Omega_u} (f_q(x,u))_i \tag{10}$$

$$\underline{l}_q^{jk} = \min_{x \in X^{jk}, u \in \Omega_u} l_q(x,u) \tag{11}$$

$$V_q^{jk} = V(x_{jk}, q) \tag{12}$$

$$\Delta_i V_q^{jk} = (V(x_{jk} + h e_i, q) - V(x_{jk}, q))/h \tag{13}$$

$$\Delta_{-i} V_q^{jk} = (V(x_{jk}, q) - V(x_{jk} - h e_i, q))/h \tag{14}$$

where $(\cdot)_i$ denote the i:th vector component of $(\cdot)$. For $A \subset \mathbf{R}^2 \times Q$, define the index set

$$I(A) = \{(j,k,q) |\ j,k \in \mathbf{Z}, q \in Q, (x_{jk}, q) \in A\} \tag{15}$$

One possible finite approximation of (4)-(6) is then given by

$$0 \le (\lambda_q^{jk})_1 + (\lambda_q^{jk})_2 + \underline{l}_q^{jk} \qquad (j,k,q) \in I(Y \backslash Y_M) \tag{16}$$

$$(\lambda_q^{jk})_{|i|} \le (\underline{f}_q^{jk})_{|i|} \Delta_i V_q^{jk} \qquad i \in \{-2,-1,1,2\},\ (j,k,q) \in I(Y \backslash Y_M) \tag{17}$$

$$(\lambda_q^{jk})_{|i|} \le (\overline{f}_q^{jk})_{|i|} \Delta_i V_q^{jk} \qquad i \in \{-2,-1,1,2\},\ (j,k,q) \in I(Y \backslash Y_M) \tag{18}$$

$$\begin{aligned} 0 \le\ & V_{\nu(x_{jk},q,\mu)}^{jk} - V_q^{jk} \\ & + s(x_{jk}, q, \nu(x_{jk}, q, \mu)) \end{aligned} \qquad (j,k,q) \in I(Y \backslash Y_M),\ \mu \in \Omega_\mu \tag{19}$$

$$0 \ge V_q^{jk} \qquad (j,k,q) \in I(Y_M) \tag{20}$$

where $\lambda_q^{jk} \in \mathbf{R}^2$ for $(j,k,q) \in I(Y \backslash Y_M)$.

The constraints (16)-(18) form a combination of backward and forward difference approximations of (4) where the variable λ_q^{jk}, whose i:th component is an approximation of $(\partial V_q^{jk}/\partial x_i)(f_q)_i$, is used to preserve the lower bound property of the continuous inequality.

For $x = x_{jk} + \theta_1 h e_1 + \theta_2 h e_2$, where $0 \le \theta_i \le 1$, define the interpolating function

$$\begin{aligned} V(x,q) = (1-\theta_1)(1-\theta_2) V_q^{jk} + \theta_1 (1-\theta_2) V_q^{(j+1)k} \\ + (1-\theta_1)\theta_2 V_q^{j(k+1)} + \theta_1 \theta_2 V_q^{(j+1)(k+1)} \end{aligned} \tag{21}$$

THEOREM 1—DISCRETIZATION IN $\mathbf{R}^2$
Define Q, Ω_u, Ω_μ, f_q, ν, Y, and Y_M as in Prop. 1. With definitions (7)–(15), and (21), if there exist $V_q^{jk} \in \mathbf{R}$ for $(j,k,q) \in I(Y)$ and $\lambda_q^{jk} \in \mathbf{R}^2$ for $(j,k,q) \in I(Y \backslash Y_M)$ that satisfy (16)–(20) then

$$\int_{t_0}^{t_M} l_{q(t)}(x(t),u(t))dt + \sum_{k=1}^{M} s(x(t_k),q(t_{k-1}),q(t_k)) \geq V(x(t_0),q(t_0))$$

for every solution to (1) that is contained in Y with $(x,q)(t_M) \in Y_M$. □

Remark 1. Any function that meets the constraints, even the trivial choice $V(x,q) = 0$, is a lower bound on the true cost. Thus, to yield useful bounds, $V(x,q)$ needs to be maximized subject to (16)–(20). The maximization could be carried out in either several points in Y simultaneously (by maximizing the sum of the value function in several points $(x_{jk},q) \in Y$) or in one point $(x_0,q_0) \in Y$.

For the discretized problem, different choices of maximization criteria may lead to different results and it would be interesting to construct an example where this difference is significant. Experience from examples shows, however, that the difference between the results of a single-point and a multi-point maximization is often small, making it possible to compute the value function in a large subset of Y solving *one* LP.

Remark 2. The restriction $(x,q)(t) \in Y$ in the optimal control problem is essential. It may happen that for some initial states x_0 there exist no admissible solutions inside X. Then the maximization of $V(x_0,q_0)$ can lead to arbitrarily large values.

Remark 3. The theorem is easily extended to $\mathbf{R}^n$. Define $\mathbf{j} = (j_1,j_2,\ldots,j_n)$ and exchange jk for the multi-index $\mathbf{j}$ in the above inequalities. The limits of all summations and enumerations should also be adjusted. Section 6 shows an example in $\mathbf{R}^3$.

Proof. Assume that $x \in X^{jk}$. Noting that $\Delta_1 V_q^{jk} = \Delta_{-1} V_q^{(j+1)k}$, $\Delta_2 V_q^{jk} = \Delta_{-2} V_q^{j(k+1)}$, the inequalities (16)–(18) taken at grid points jk, $j(k+1)$, $(j+1)k$, and $(j+1)(k+1)$ give

$$0 \leq f_{q1}(x,u)\Delta_1 V_q^{jk} + f_{q2}(x,u)\Delta_2 V_q^{jk} + l_q(x,u) \tag{22}$$

$$0 \leq f_{q1}(x,u)\Delta_1 V_q^{j(k+1)} + f_{q2}(x,u)\Delta_2 V_q^{jk} + l_q(x,u) \tag{23}$$

$$0 \leq f_{q1}(x,u)\Delta_1 V_q^{jk} + f_{q2}(x,u)\Delta_2 V_q^{(j+1)k} + l_q(x,u) \tag{24}$$

$$0 \leq f_{q1}(x,u)\Delta_1 V_q^{j(k+1)} + f_{q2}(x,u)\Delta_2 V_q^{(j+1)k} + l_q(x,u) \tag{25}$$

The gradient of V is given by

$$\frac{\partial V_q}{\partial x} = \begin{bmatrix} (1-\theta_2)\Delta_1 V_q^{jk} + \theta_2 \Delta_1 V_q^{j(k+1)} \\ (1-\theta_1)\Delta_2 V_q^{jk} + \theta_1 \Delta_2 V_q^{(j+1)k} \end{bmatrix}^T$$

and thus, adding (22)–(25) weighted with $(1-\theta_1)(1-\theta_2)$, $(1-\theta_1)\theta_2$, $\theta_1(1-\theta_2)$, and $\theta_1\theta_2$ respectively proves that (4) is met for x. The inequality (5) holds since V is a convex combination of grid points that all meet (19), and (6) is the same condition as (20). □

Note that the minimization/maximization in (9)–(11) is in general not convex. However, Theorem 1 can be applied with any upper and lower bounds on f_q and l_q and such bounds are often easy to obtain.

Also note a special case for which the burden of the local optimizations in Theorem 1 is lightened: if $f_q(x,u) = h_q(x) + g_q(x)u$ and $l_q(x,u) = o_q(x) + m_q(x)u$ while $\Omega_u = [-1,1]$, then u can be entirely eliminated from (16)–(18) by replacing $\underline{f}_q^{jk}$, $\overline{f}_q^{jk}$, and $\underline{l}_q^{jk}$ with $\underline{h}_q^{jk} \pm \underline{g}_q^{jk}$, $\overline{h}_q^{jk} \pm \overline{g}_q^{jk}$, and $\underline{o}_q^{jk} \pm \underline{m}_q^{jk}$ respectively. This will double the set of equations (16)–(18), but the functions h_q, g_q, o_q, and m_q are optimized over X^{jk} solely.

5. Computing the Control Law

Provided that the lower bound V is a good enough approximation of the optimal cost, the optimal feedback control law can be calculated as

$$\begin{cases} u(x,q) = \operatorname{argmin}_{\hat{u}\in\Omega_u} \left\{ \frac{\partial V}{\partial x}(x,q) f_q(x,\hat{u}) + l_q(x,\hat{u}) \right\} \\ \mu(x,q) = \operatorname{argmin}_{\hat{\mu}\in\Omega_\mu | x\in S_{q,\nu}} \left\{ V(x,\nu) + s(x,q,\nu) \right\} \end{cases} \tag{26}$$

where $\nu = \nu(x,q,\hat{\mu})$. Note that the discrete input, μ, is chosen such that switching occur whenever there exist a discrete mode for which the value function has a lower value than the cost of the value function for the current mode minus the cost for switching there.

Consider the true optimal value function, $V^\star$. For those (x,q,r) where the optimal trajectory requires mode switching, the inequality (4) will turn to equality i.e. $V^\star(x,q) = V^\star(x,r) + s(x,q,r)$. A consequence of this is that for (26) to describe correct switching between the modes, $s(x,q,q)$ has to be defined as $s(x,q,q) = \varepsilon > 0$ (rather than the natural choice $s(x,q,q) = 0$). For $V^\star$, the proper control law is achieved as ε approaches 0. A small value of ε suffices, however, for numerical computations.

In practice, it is suitable to discretize u into $\hat{\Omega}_u = \{u_1, u_2, \ldots, u_a\} \subset \Omega_u$. Then for each grid point, x^{jk}, the problem is to find $u_q^{jk} \in \hat{\Omega}_u$ and $\mu_q^{jk} \in \Omega_\mu$

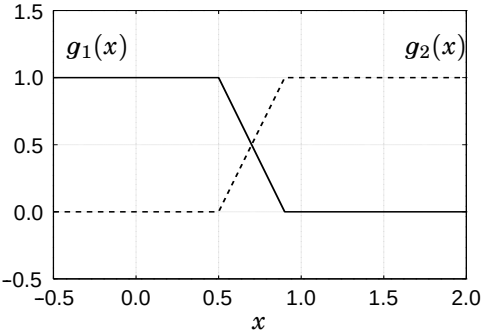


Figure 1. Gear profiles for the truck.

off-line that minimize (26). On-line, during control, $u(x,q)$ and $\mu(x,q)$ are obtained by multilinear interpolation analogously to (21).

The choice of $\hat{\Omega}_u$ may be crucial and is often a trade off between speed of computations and how close to optimal the result will be. For time optimal control problems where u enters affinely, however, the control signal only assumes its extremal values.

6. Numerical Example — A Truck With a Flexible Transmission

The applicability of the theory is here illustrated by an example with three continuous states. See [Hedlund and Rantzer, 1999] for additional examples. Consider the system

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \frac{1}{m}(-cx_2 + kx_3) \\ \dot{x}_3 = -x_2 + \frac{g_q(x_2)}{k}u, \qquad q = 1,2 \qquad -0.1 \le u \le 1.1 \end{cases} \tag{27}$$

The three continuous states of the system could be seen as position (x_1) and velocity (x_2) of a truck, and the rotational displacement of its transmission shaft (x_3). There are two discrete modes corresponding to different gears of the truck; the input throttle, u, is weighted by $g_q(x)$ that represents the efficiency of gear number q. The weighting functions are plotted in Fig. 1.

All the constants (the mass of the car, m, the frictional damping, c, and the spring constant of the transmission shaft, k) are set to 1.

The objective is to bring Eq. (27) to $Y_M = \{(0,0)\}$ in minimum time. Torque losses when using the clutch calls for an additional penalty for gear

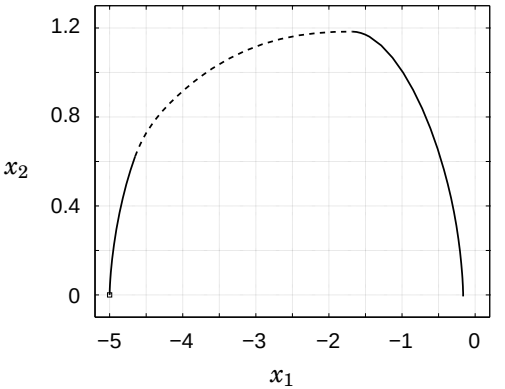


Figure 2. Phase portrait of x_1 and x_2 under simulation. The solid line shows where gear number one has been used, the dashed line shows the second gear. The initial point is marked with a square.

changes. Thus, the terms of (3) have been chosen as $l_1(x,u) = l_2(x,u) = 1$, $s(x,1,2) = s(x,2,1) = 0.8$.

Since it is difficult to visualize the three dimensional value function, it is not shown here. A feedback control law is derived from the value function, however, and results from simulations using this law are shown in Fig. 2.

With the current cost function, it is obvious that whenever a gear switch is required, it is optimal to switch at the speed of equal efficiency between the gears ($x_2 = 0.7$). This action can be noted in the figure when switching from the first gear to the second. The switch back to the first gear during the deceleration phase, however, occurs in the simulation at a much higher (non optimal) speed. This is a reasonable approximation error though, since the deceleration power is small ($u = -0.1$). The difference in cost depending on how early the gear switch is made, is low compared to the total cost.

Figure 3 shows how the rotational displacement of the transmission shaft varies with u. The spring tension builds up during the acceleration phase (approximately. $0 \leq t \leq 4.3$) and is then released.

An upper bound is obtained by integrating the cost along the simulated trajectory, starting in $x_i = (-5, 0, 0)^T$, $q_i = 1$, is 8.5. The lower bound given by the value function is 7.9.

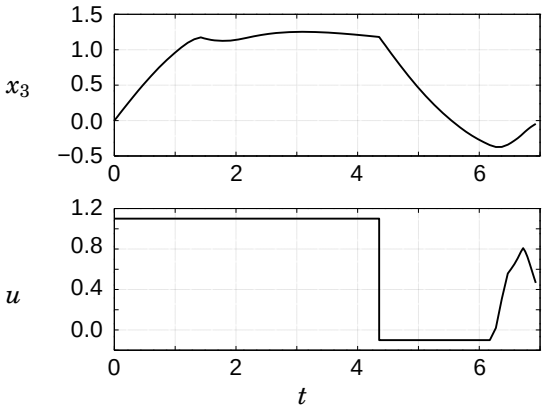


Figure 3. Plot of spring tension (x_3) and the continuous valued control signal (u).

7. Conclusion

This paper presented an extended version of the Hamilton-Jacobi-Bellman inequality to be used for optimal control of hybrid systems. The extended version constitutes a successful marriage between computer science and control theory, containing pure discrete dynamic programming as well as pure continuous dynamic programming as special cases.

The extended HJB inequality, that gives a lower bound on the value function, was discretized to a finite, computer solvable LP that preserves the lower bound property. Based on the value function, an approximation of the optimal control feedback law was derived.

A problem with DP is the "curse of dimensionality", an expression that was coined by Richard Bellman, the inventor of this method. Since the cost for a family of trajectories is computed (rather than a single trajectory as in the Pontryagin Maximum Principle), the problem grows exponentially in the number of states.

The advantage with this method, however, is its applicability and ease of use for low dimension systems. The discretization method presented in this paper allows problems with up to three continuous states on a 336MHz Ultra Sparc II.

A set of MATLAB commands has been compiled by the authors to make it easy to test the above methods and implement the examples. The LP solver that is used is "PCx", developed by the Optimization Technology Center, Illinois. The MATLAB commands and a manual of usage are available free of charge upon request from the authors.

8. Acknowledgments

This work has been supported by the European Commission under ESPRIT LTR-Project 28104, called H^2C.

Paper B

Duality Between Cost and Density in Optimal Control

Anders Rantzer and Sven Hedlund

Abstract

A theorem on duality between cost functions and density functions in optimal control is derived using the Hahn-Banach theorem. The result puts focus on convexity aspects in control synthesis and the recent theory of almost global stability. In particular, it gives a new proof that existence of a density function is both necessary and sufficient for almost global stability in a nonlinear system.

Keywords Density function, duality, stability, optimal control

A. Rantzer and S. Hedlund: "Duality between cost and density in optimal control", submitted to *Conference on Decision and Control 2003*.

1. Introduction

The idea of duality between cost and flow has old roots. In fact, a non-linear problem of optimal transportation stated by G. Monge in 1781 was converted into convex optimization by [Kantorovich, 1942] and inspired much of the later developments in the theory of convex duality. See [Rachev and Rüschendorf, 1998]. Kantorovich later received the Nobel price for related work in mathematical economics.

The ideas were introduced in the context of optimal control by [Young, 1969] using the concept of generalized flow. For later work, see [Vinter, 1993]. More recently, [Rantzer, 2001] introduced the concept of density function as a tool for verification of almost global stability in non-linear systems. The relation to duality theory was then briefly discussed.

The new stability concept has a remarkable convexity property in the context of control synthesis. This was explored for numerical computations in [Rantzer and Parrilo, 2000] and for smooth transitions between different nonlinear controllers in [Rantzer and Ceragioli, 2001].

The purpose of the present paper is to establish the duality between cost functions and density functions in a more rigorous manner. The main result is stated and discussed in section 2. The next two sections are devoted to the proof. The main duality argument is given section 3, while the relation to optimal control is established in section 4.

2. Main Result

Let $f_i \in \mathcal{C}^1(\mathbf{R}^n, \mathbf{R}^n)$, $l_i \in \mathcal{C}(\mathbf{R}^n, \mathbf{R}^m)$ with $l_i \geq 0$ for $i = 1, \ldots, M$. Let $\Gamma, X \subset \mathbf{R}^n$ be open bounded sets with C^1 boundary and $\overline{\Gamma} \subset X$. Suppose that f_i points strictly inwards on the boundary of X and the same on the boundary of Γ. Introduce $\mathcal{U}$ as the set of all non-negative $(u_1, \ldots, u_M) \in \mathcal{C}^1(X, \mathbf{R}^M)$ with $u_1(x) + \cdots + u_M(x) \equiv 1$. The solution of $\dot{x} = \sum_i u_i(x) f_i(x)$, $x(0) = x_0$ is denoted $\phi_u(x_0, t)$. Let

$$V_u(x) = \sum_i \int_0^\infty u_i\left(\phi_u(x,t)\right) l_i\left(\phi_u(x,t)\right) dt$$
$$V^*(x) = \inf_{u \in \mathcal{U}} V_u(x)$$

The main theorem can now be stated as follows:

THEOREM 1
Consider X, f_i, l_i, $\mathcal{U}$, ϕ_u and V^* as above. Let $l_i > 0$ outside Γ and define

$\psi \in \mathcal{C}(\overline{X})$ with $\psi > 0$. Then

$$\sup_V \int_X \psi(x)V(x)dx = \int_X \psi(x)V^*(x)dx = \inf_{\rho_i} \sum_{i=1}^{M} \int_X l_i(x)\rho_i(x)dx$$

where sup is taken over non-negative $V \in \mathcal{C}^1(\mathbf{R}^n)$ such that for $i = 1, \ldots, M$

$$\nabla V(x) \cdot f_i(x) + l_i(x) > 0 \qquad x \in X \setminus \Gamma \tag{1}$$
$$V(x) = 0 \qquad x \in \Gamma \tag{2}$$

and inf is taken over $\rho_i \in \mathcal{C}_0^1(\overline{X})$ with $\rho_i > 0$ in X and

$$\sum_{i=1}^{M} \nabla \cdot (f_i(x)\rho_i(x)) > \psi(x) \qquad x \in X \setminus \Gamma \tag{3}$$

Moreover, $u := (\rho_1, \ldots, \rho_M)/(\sum_i \rho_i)$ is an element in $\mathcal{U}$ and

$$\int_X \psi(x)V_u(x)dx < \sum_{i=1}^{M} \int_X l_i(x)\rho_i(x)dx$$

□

Before giving the proof in the following sections, we make a couple of remarks.

The Case of No Control Variable (M=1) In this case, the value of the integral

$$\int_X \psi(x)V^*(x)dx \tag{4}$$

is interesting as a stability indicator. A finite value of the integral means that $V^*(x) = \int_0^\infty l(x(t))dt$ is finite for almost all $x \in X \setminus \Gamma$. For all these initial states, the trajectory must approach Γ as $t \to \infty$. Hence, Theorem 1 proves existence of non-negative $\rho \in \mathcal{C}_0^1(\overline{X})$ such that

$$\nabla \cdot (f(x)\rho(x)) > \psi(x) > 0 \qquad x \in X \setminus \Gamma$$

Conversely, if such a ρ exists, then the theorem shows that (4) is finite and almost all trajectories eventually approach Γ.

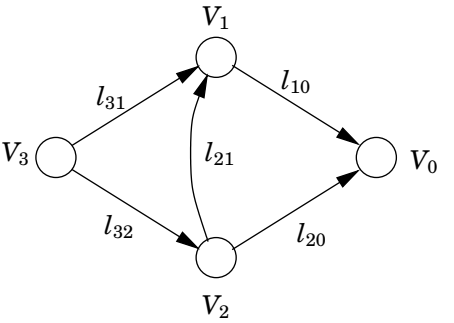


Figure 1. The products produced in nodes 1-3 should be transported to the consumer in node 0 while minimizing the transportation cost.

Control Synthesis by Convex Optimization It should be noted that the minimization corresponding to the infimum expression in Theorem 1 is a problem of convex optimization. In fact, every multiple $(\rho_1, \ldots, \rho_M)$ that solves the divergence inequality not only gives an upper bound on the optimal value $\int_X \psi(x)V^*(x)dx$, but also corresponds to a control law achieving this upper bound. This can be viewed as the reason behind the previously mentioned convexity property of density functions in control synthesis.

Comparison to a Discrete Transportation Problem It is natural to compare Theorem 1 to the standard linear programming solution to the discrete transportation problem illustrated in Figure 1. Such problems have been studied extensively since the 1940's [Hitchcock, 1941; Ford and Fulkerson, 1962]. Some product is produced with unit rate in each of the three nodes 1-3 and is consumed in node 0. The cost for shipping the product between node i and node j is given by the number l_{ij}. It is well known that the minimal total transportation cost can be found by solving the following linear programming problem.

$$\begin{aligned} \text{Maximize} \quad & V_1 + V_2 + V_3 - 3V_0 \\ \text{subject to} \quad & V_3 - V_1 \leq l_{31} \\ & V_3 - V_2 \leq l_{32} \\ & \vdots \\ & V_2 - V_0 \leq l_{20} \end{aligned}$$

Note that there is one variable V_i for each node and one inequality constraint for each path connecting two nodes. For every solution to the in-

equality constraints, the number $V_i - V_0$ provides a lower bound on the cost for shipping products with unit rate from node i to node 0. The expression $V_1 + V_2 + V_3 - 3V_0$ therefore gives a lower bound on the total transportation cost.

A dual linear programming problem can be stated as follows.

$$\begin{aligned} \text{Minimize} \quad & l_{31}\rho_{31} + l_{32}\rho_{32} + l_{21}\rho_{21} + l_{10}\rho_{10} + l_{20}\rho_{20} \\ \text{subject to} \quad & \rho_{31}, \ldots, \rho_{20} \geq 0 \\ & \rho_{31} + \rho_{32} \geq 1 \\ & -\rho_{31} - \rho_{21} + \rho_{10} \geq 1 \\ & -\rho_{32} + \rho_{21} + \rho_{20} \geq 1 \end{aligned}$$

For each path connecting two nodes, the variable ρ_{ij} can be interpreted as the transportation density from node i to node j. There is one constraint for each node stating that the total production in this node is at least as big as the assigned value.

3. A Min-Max Inequality

An inequality relating the supremum and the infimum in Theorem 1 will next be proved as a separate statement.

LEMMA 1
Let $f_i \in \mathcal{C}^1(\mathbf{R}^n, \mathbf{R}^n)$, $l_i \in \mathcal{C}(\mathbf{R}^n, \mathbf{R}^m)$ with $l_i \geq 0$ for $i = 1, \ldots, M$. Let $\Gamma \subset X$ be open and bounded subsets of $\mathbf{R}^n$ with C^1 boundary. Suppose that f_i points strictly inwards on the boundary of X and the same on the boundary of Γ. Let $l_i > 0$ outside Γ and define $\psi \in \mathcal{C}(\overline{X})$ with $\psi > 0$. Then

$$\sup_V \int_X \psi(x) V(x) dx \geq \inf_{\rho_i} \sum_{i=1}^{M} \int_X l_i(x) \rho_i(x) dx \tag{5}$$

where supremum is taken over all non-negative $V \in \mathcal{C}^1(\mathbf{R}^n)$ satisfying

$$\nabla V(x) \cdot f_i(x) + l_i(x) > 0 \qquad x \in X \setminus \Gamma \tag{6}$$

$$V(x) = 0 \qquad x \in \Gamma \tag{7}$$

and infimum is taken over all $\rho_i \in \mathcal{C}_0^1(\overline{X})$ with $\rho_i > 0$ in X and

$$\sum_{i=1}^{M} \nabla \cdot (f_i(x)\rho_i(x)) > \psi(x) \qquad x \in X \setminus \Gamma \tag{8}$$

□

Proof. Define two subsets of $K = \mathbf{R} \times \mathcal{C}(\overline{X})$:

$$K_1 = \left\{ \left(-\sum_{i=1}^{M} \int_X l_i \rho_i dx + \gamma, \sum_{i=1}^{M} \nabla \cdot (f_i \rho_i) - \psi \right) \Bigg| \rho_i \in \mathcal{C}_0^1(\overline{X}),\ \rho_i > 0 \text{ in } X \right\}$$

$$K_2 = \{(z,h) \in K \mid z \geq 0,\ h(x) > 0 \text{ for } x \in X \setminus \Gamma\}$$

For the following five statements, we will prove the implications **I**⇔**II**⇔**III** ⇔**IV**⇒**V**.

I The number γ is not larger than the right hand side of (5).

II K_1 contains no interior point of K_2.

III There exists $k^\star \in K^\star, k^\star \neq 0$ such that

$$\sup_{k_1 \in K_1} \langle k^\star, k_1 \rangle \leq \inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle, \tag{9}$$

where $K^\star$ is the dual space of K, i.e. $K^\star = \mathbf{R} \times \mathcal{C}(\overline{X})^\star$

IV There exists a nonzero pair (a, ϕ) where $a \geq 0$ is a number and $\phi \geq 0$ is a measure of bounded variation on $\overline{X}$, vanishing inside Γ, such that $\langle \phi, \psi \rangle := \int_X \psi(x) d\phi(x) \geq a\gamma$ and

$$a l_i + \nabla \phi \cdot f_i \geq 0 \qquad \text{in } X \text{ for } i = 1, 2, \ldots, M \tag{10}$$

The derivative $\nabla \phi$ is interpreted in the sense of distributions.

V The number γ is not larger than the left hand side of (5).

The equivalence **I**⇔**II** is trivial once it is noted that $(z,h) \in K_2$ is an interior point if and only if $z > 0$ and $h(x) > 0$ for $x \in \overline{X}$. The second equivalence **II**⇔**III** holds because of the following separation property of convex sets [Luenberger, 1969; Rudin, 1991]:

Let K be a normed vector space and denote its dual $K^\star$. Let K_1 and K_2 be convex sets in K such that K_2 has interior points and K_1 contains no interior point of K_2. Then there is a closed hyperplane separating K_1 and K_2; i.e., there is a $k^\star \in K^\star$, $k^\star \neq 0$ such that

$$\sup_{k_1 \in K_1} \langle k^\star, k_1 \rangle \leq \inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle, \tag{11}$$

To show that **III**⇔**IV**, let $k^\star = (a, \phi) \in K^\star = \mathbf{R} \times \mathcal{C}(\overline{X})^\star$. The space $\mathcal{C}(\overline{X})^\star$ is the set of measures of bounded variation and support in $\overline{X}$ [Dunford and Schwartz, 1958, page 262]. Expand the right hand side of (9) to

$$\inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle = \inf_{(z,h) \in K_2} \{az + \langle \phi, h \rangle\} \tag{12}$$

The right hand side is equal to zero if and only if a and ϕ are both non-negative and $\phi = 0$ in Γ. Otherwise it is $-\infty$.

The left hand side of (9) can be expanded to

$$\begin{aligned}
&\sup_{k_1 \in K_1} \langle k^\star, k_1 \rangle \\
&= \sup_{\rho_i} \Big\{ a\big(-\sum_i \int_X l_i \rho_i dx + \gamma\big) + \big\langle \phi, \sum_i \nabla \cdot (f_i \rho_i) - \psi \big\rangle \Big\} \\
&= \sup_{\rho_i} \Big\{ \sum_i \big\langle -a l_i - \nabla \phi \cdot f_i, \rho_i \big\rangle \Big\} + a\gamma - \langle \phi, \psi \rangle
\end{aligned}$$

The supremum is taken over $\rho_i \in \mathcal{C}^1(\mathbf{R}^n)$ with support in $\overline{X}$ and $\rho_i > 0$ in X. The value is equal to $a\gamma - \langle \phi, \psi \rangle$ if and only if (10) holds, otherwise it is $+\infty$. The statement **III** is thus equivalent to **IV**.

It remains to prove the implication from **IV** to **V**. Hence, assume existence of a non-zero pair (a, ϕ) as in **IV**. Note that ϕ can be identified with a distribution of order zero on $\mathbf{R}^n$, which is identically zero outside $\overline{X}$. If $a = 0$, define $\bar{a} = \bar{\varepsilon}$ and $\bar{\gamma} = \gamma$. If $a > 0$, define $\bar{a} = a(1 + \bar{\varepsilon})$ and $\bar{\gamma} = \gamma/(1 + 2\bar{\varepsilon})$. In both cases, for sufficiently small $\bar{\varepsilon} > 0$,

$$0 < \bar{a} l_i + \nabla \phi \cdot f_i \quad \text{ in } \mathbf{R}^n \setminus \Gamma \qquad\qquad \langle \phi, \psi \rangle > \bar{a}\bar{\gamma} \tag{13}$$

Here, the assumption that all f_i points strictly inwards on the boundary of X is critical to guarantee that $\nabla \phi \cdot f_i \geq 0$ on the boundary of X.

Note that ϕ cannot be singular on the boundary of Γ, due to the first inequality of (13) in combination with the assumption that f_i points strictly inwards on the boundary of Γ. Hence, one can write $\phi = \sum_{i=1}^{\infty} \phi_i$ where each ϕ_i is zero within a distance ε^i from Γ. Each of the terms can be regularized by convolution with a smooth non-negative function having support in a ball of sufficiently small radius and integral equal to one. In this way, ϕ can be approximated by a smooth function function ϕ_0, vanishing inside Γ and satisfying

$$0 < \bar{a} l_i + \nabla \phi_0 \cdot f_i \quad \text{ in } \mathbf{R}^n \setminus \Gamma \qquad\qquad \langle \phi_0, \psi \rangle > \bar{a}\bar{\gamma}$$

Finally, $V = \phi_0/\bar{a} \in \mathcal{C}^1(\mathbf{R}^n)$ gives (6)-(7) and

$$\int_X \psi(x) V(x) dx > \bar{\gamma}$$

Condition **V** follows and the proof is complete. □

4. Proof of the Main Theorem

Two more lemmata are needed before completing the proof of Theorem 1.

LEMMA 2
Suppose that $X \subset \mathbf{R}^n$ is open and bounded and that $\Gamma \subset X$ is open. Given $f \in \mathcal{C}(X)$ and $l \in \mathcal{C}(X)$, assume that $l(x) \geq 0$ with strict inequality outside Γ. If $V \in \mathcal{C}^1(X)$, $V(x) = 0$ for $x \in \Gamma$ and

$$\nabla V(x) \cdot f(x) + l(x) \geq 0 \qquad\qquad x \in X \setminus \Gamma$$

then $V(x_0) \leq \int_0^\infty l\left(x(t)\right) dt$ when $\dot{x} = f(x)$, $x(0) = x_0$ and $x(t) \in X$ for all t. □

Proof. If $\int_0^\infty l\left(x(t)\right) dt < \infty$, then there exists a sequence $0 < t_1 < t_2 < \ldots$ where $t_i \to \infty$ as $i \to \infty$ such that $\lim_{i\to\infty} l\left(x(t_i)\right) = 0$. It follows that $x(t_i) \to \Gamma$ as $i \to \infty$ and

$$\begin{aligned} V(x_0) - V(x(t_i)) &= -\int_0^{t_i} \frac{d}{dt} V(x(t)) dt \\ &= -\int_0^{t_i} \frac{\partial V}{\partial x} f(x(t)) dt \leq \int_0^\infty l(x(t)) dt \end{aligned}$$

In the limit as $i \to \infty$ we get the desired inequality. □

LEMMA 3
Let $X \subset \mathbf{R}^n$ be open and bounded. Given $f \in \mathcal{C}^1(X, \mathbf{R}^n)$, suppose that X and the subset Γ are both invariant under the dynamics $\dot{x} = f(x)$. Suppose that $l \in \mathcal{C}(X)$, $\rho \in \mathcal{C}^1(X)$ and $\psi \in \mathcal{C}(X)$ are non-negative. Define $V(x) = \int_0^\infty l\left(\phi(x,t)\right) dt$. Then

$$\int_X \psi(x) V(x) dx \leq \int_X l(x) \rho(x) dx$$

provided that $\nabla \cdot (f\rho)(x) > \psi(x)$ for $x \in X \setminus \Gamma$. □

The proof of Lemma 3 uses the following version of Liouville's theorem [Rantzer, 2001]:

PROPOSITION 1
Let $f \in C^1(X, \mathbf{R}^n)$ and let $\rho \in C^1(X)$ be non-negative and integrable. For a measurable set Z, assume that $\phi_\tau(Z) = \{\phi_\tau(x) \mid x \in Z\}$ is a subset of X for all τ between 0 and t. Then

$$\int_{\phi(Z,t)} \rho(x)dx - \int_Z \rho(z)dz = \int_0^t \int_{\phi(Z,\tau)} [\nabla \cdot (f\rho)](x)dxd\tau$$

□

Proof of Lemma 3. Consider $T > 0$ and a piecewise constant $l(x) = \sum_i l_i \chi_i(x)$, where χ_i is the characteristic function of the set $X_i \subset X$. Then

$$\begin{aligned}
\int_X l(x)\rho(x)dx &\geq \sum_i l_i \left(\int_{X_i} \rho(x)dx - \int_{\phi(X_i,-T)} \rho(x)dx \right) \\
&= \sum_i l_i \int_0^T \int_{\phi(X_i,-t)} \nabla \cdot (f\rho)(z)dzdt \\
&> \sum_i l_i \int_0^T \int_{\phi(X_i,-t)} \psi(z)dzdt \\
&= \sum_i l_i \int_0^T \int_X \chi_i(\phi(z,t))\psi(z)dzdt \\
&= \int_X \psi(z) \int_0^T l(\phi(z,t))dtdz
\end{aligned}$$

This proves the desired inequality for piecewise constant l. Non-strict inequality follows by continuity for arbitrary continuous l. □

Proof of Theorem 1 Let γ^* be the value of the infimum. Define $\rho_i \in C_0^1(\overline{X})$, $i = 1, \ldots, M$ such that (3) holds, $\rho_i > 0$ in X and

$$\sum_{i=1}^M \int_X l_i(x)\rho_i(x)dx < \gamma^* + \varepsilon$$

for some $\varepsilon > 0$. Define $u \in C^1(X)$ and $f \in C^1(X)$ according to

$$\begin{aligned}
\rho(x) &= \sum_i \rho_i(x) \\
u_i(x) &= \frac{\rho_i(x)}{\rho(x)} \\
f(x) &= \sum_i u_i(x) f_i(x)
\end{aligned}$$

Then

$$\nabla \cdot (f\rho)(x) = \sum_i \nabla \cdot (f_i \rho_i)(x) > \psi(x) \qquad\qquad x \in X$$

and for every solution to the equation $\dot{x} = f(x)$

$$0 < \frac{\psi(x)}{\rho(x)} < \frac{\nabla \cdot (f\rho)}{\rho} = \nabla \cdot f + \frac{\nabla \rho}{\rho} \cdot f = \nabla \cdot f + \nabla(\log \rho) \cdot f$$

$$0 < (\nabla \cdot f)(x(t)) + \frac{d}{dt} \log \rho(x(t))$$

By continuity of $\nabla \cdot f$ on the compact set $\overline{X}$, there is a constant C such that $\nabla \cdot f(x) \leq C$ for $x \in X$. Hence

$$-C < \frac{d}{dt} \log \rho(x(t)) \qquad\qquad \rho(x(t)) > e^{-Ct} \rho(x(0))$$

This shows that the trajectory can not reach the boundary of X, where $\rho = 0$, but must stay in X for all $t \geq 0$. Hence by Lemma 3

$$\int_X \psi(x) V_u(x) dx \leq \sum_{i=1}^{M} \int_X l_i(x) \rho_i(x) dx < \gamma^* + \varepsilon$$

The choice of $\varepsilon > 0$ was arbitrary, so

$$\int_X \psi(x) V^*(x) dx \leq \gamma^*$$

For an opposite bound, Let γ_* be the value of the supremum. Define $V \in C^1(X)$ non-negative, satisfying (1)-(2) and

$$\gamma_* - \varepsilon < \int_X \psi(x) V(x) dx$$

Then by Lemma 2

$$\gamma_* - \varepsilon < \int_X \psi(x) V^*(x) dx$$

In the limit as $\varepsilon \to 0$ we get

$$\gamma_* \leq \int_X \psi(x) V^*(x) dx \leq \gamma^*$$

Finally, Lemma 1 proved that $\gamma^* \leq \gamma_*$, so the proof is complete. □

Paper C

Duality in Hybrid Optimal Control

Sven Hedlund and Anders Rantzer

Abstract

It has lately been reported that a lower bound of the value function of a hybrid optimal control problem can be computed via a linear program. Moreover, a dual of this optimization problem has been formulated via general ideas from duality in transportation problems.

Paper B has shown that the lower bound of the value function is tight for continuous systems and that there is no gap between the dual optimization problems. This paper extends the main result of Paper B to the hybrid system formulation.

This paper also tries to convey an intuitive understanding of the two different optimization approaches via a thorough discussion around simple examples.

Keywords Hybrid systems, optimal control, dynamic programming, linear program, duality.

1. Introduction

An optimal control problem in general contains a dynamical system and a cost function to be minimized by the choice of control law. In this paper, the performance of the control law is measured as a weighted sum of the cost for trajectories starting in a certain set of initial states. Previous papers have shown that the resulting optimization problem can be approached in two different ways, both approaches resulting in a linear program.

One approach gives a lower bound on the performance and the corresponding decision variables are a lower bound of the value function, see [Hedlund and Rantzer, 2002] for treatment of hybrid systems. The other approach, a dual formulation of the first approach, uses decision variables that are closely related to the optimal control law, see [Hedlund and Rantzer, 2000] for a hybrid system version.

It has recently been shown for continuous systems that the lower bound is tight and that there is no gap between the dual optimization problems, see Paper B of this thesis, and the same can be shown to hold for for discrete systems. Moreover, the proof for discrete systems can be built with the same components, lemmas and propositions, as for the continuous case. Following the same procedure for hybrid systems, merging the continuous lemmas with their discrete counterparts, this paper proves a major part of a corresponding duality theorem for hybrid systems.

The dual formulations of the minimum cost problems in this paper can be interpreted as optimizing mass flows and the idea of duality between cost and flow has old roots. In fact, a non-linear problem of optimal transportation stated by G. Monge in 1781 was converted into convex optimization by [Kantorovich, 1942] and inspired much of the later developments in the theory of convex duality. See [Rachev and Rüschendorf, 1998]. Kantorovich later received the Nobel price for related work in mathematical economics.

The ideas were introduced in the context of optimal control by [Young, 1969] using the concept of generalized flow. For later work, see [Vinter, 1993].

The outline of this paper is as follows. In Section 2, an optimal control problem for a discrete system is solved in two different ways. These two approaches result in two different LP formulations, one being the dual of the other. This section does not contain any new theory, but is included to introduce ideas and concepts to make way for the understanding of the theory for the more complex continuous and hybrid systems.

A more rigorous presentation of duality for discrete systems is given in a theorem in Section 3. The different parts of this theorem have their counterparts in a duality theorem for continuous systems, presented in Section 4. The main part of the above mentioned theorems are merged in

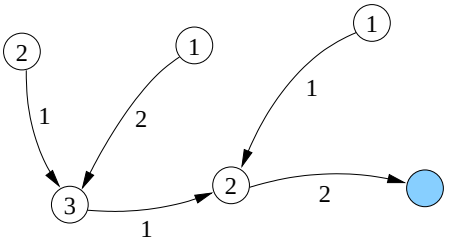


Figure 1. Graphic illustration of the transportation example. The circles represent cities and the numbers in the circles are their milk production. The colored circle is a city with a dairy. The arrows are roads and the number next to each arrow is the road tax.

Section 5 to a duality for hybrid systems.

Section 6 contains an example on the importance of the formulation of the optimal control problem, and Section 7 summarizes our results. The proofs of the duality for discrete systems and hybrid systems are presented in Section 8. (For a proof of the continuous system duality, see Paper B.)

2. Duality in a Transportation Problem

Imagine a nation with several cities and several roads that connect the cities. There are cows with a steady rate milk production close to all but one city. In the city without milk production, there is a dairy. There is one company transporting all of the milk that is produced by trucks to the dairy.

For each production city, the company has picked *one* exit road to be used by *all* transports leaving that city. There is a certain road tax, transportation cost, connected to each road.

A graph of the setting is shown in Fig. 1. The circles represent the cities, and the number in each circle is the milk production measured in trucks per day. The shaded circle is the city with a dairy. The arrows represent the roads that have been chosen for transportation and the number next to each arrow is the road tax per truck for that road.

2.1 Computation of the Total Transportation Cost

Consider the problem of computing the total transportation cost per day. This problem can be approached in two ways.

The first approach is called the “cost first approach” or “value function approach”. For each city, compute the total transportation cost per truck

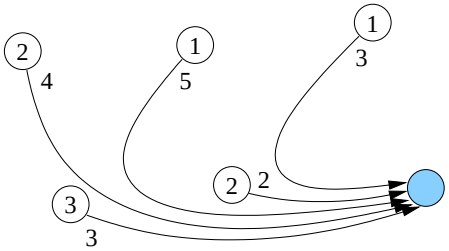


Figure 2. Transportation cost computation, value function approach. The number in each circle is the milk production in the corresponding city and the number at the beginning of each arrow is the total road tax from the city to the dairy.

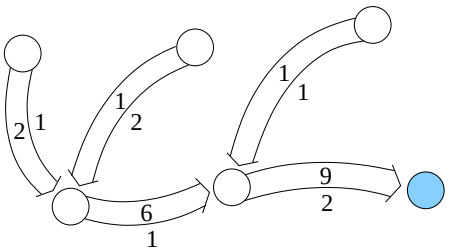


Figure 3. Transportation cost computation, density function approach. The number inside each arrow is the transport flow density, i.e. the total number of trucks per day on the road. The number next to each arrow is the road tax of the corresponding road.

from that city to the dairy, see Fig. 2. The resulting map from cities to their transportation costs is called the cost-to-go function or value function. The daily cost for the complete transport of the production in one city is the cost-to-go multiplied by its production number. The total cost for this example is $4 \cdot 2 + 5 \cdot 1 + 3 \cdot 1 + 3 \cdot 3 + 2 \cdot 2 = 29$.

The second approach is called the "flow first approach" or "density function approach". Compute the daily number of trucks on each road, i.e. for a road leaving a city, sum all the trucks entering the city plus the production in the city itself, see Fig. 3. The resulting map from cities to their outflow is called the flow density function or just density function. The daily cost for all transports from one city to its immediate successor is obtained by multiplying the flow density out from the city with its road tax. The total transportation cost for this example is $2 \cdot 1 + 1 \cdot 2 + 1 \cdot 1 + 6 \cdot 1 + 9 \cdot 2 = 29$.

2.2 Optimizing Transports for Minimum Total Cost

The company wants to reconsider their choice of transportation roads in order to minimize the total cost. This optimization problem can be solved in the spirit of either of the two approaches above.

Value Function Approach The total cost depends on the value function, which in turn depends on the choices of roads. Hence, a naive approach is to try a certain set of roads, compute the corresponding value function, compute the corresponding total cost, and then repeat the whole procedure over and over again to iteratively get a lower total cost.

Note, however, that there are some properties of the value function that are independent of road choice:

- If there is a road from city A to city B, then the value function in A can not exceed the value function in B plus the road tax from A to B.
- The value function is zero in the city with the dairy, since there is no remaining transportation to the goal.

In fact, the above properties make it possible to obtain the optimal value function (the value function corresponding to the lowest total cost) regardless of the underlying road choice. The optimal value function is the largest function (measured by the sum of the value function weighted with the production for each city) with the above two properties and it can be found by solving a linear program. This is illustrated by another simple example.

EXAMPLE 1—THE MINIMUM COST EXAMPLE, VALUE FUNCTION APPROACH. There are four cities in this example, see Fig. 4. City a, b, and c each produce one truck of milk per day. The dairy is in city d. Let V_a, V_b, V_c, and V_d denote the value function for city a, b, c, and d respectively. The aforementioned value function properties translate to

$$\begin{aligned} V_a &\leq V_b + 2 \\ V_a &\leq V_c + 4 \\ &\vdots \\ V_c &\leq V_d + 1 \end{aligned}$$

and

$$V_d = 0$$

It will be shown in Sec. 3 that the *optimal* value function is the largest function that meets the above constraints, i.e. the one maximizing

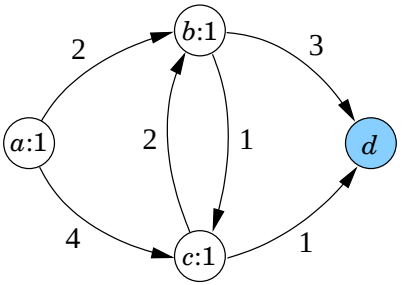


Figure 4. Graphic illustration of the minimum cost example. The letter in each circle is the "name" of the city, while the figure to the right of each colon still is the production.

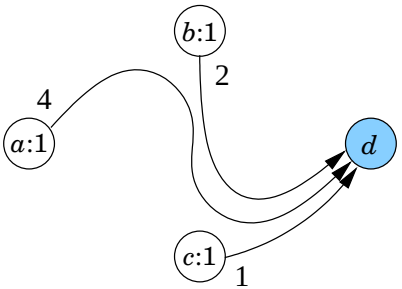


Figure 5. Optimal value function for the minimum total cost example: $V_a = 4$, $V_b = 2$, $V_c = 1$. The total cost is $1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 7$

$1 \cdot V_a + 1 \cdot V_b + 1 \cdot V_c$. Having obtained the solution to this linear program, i.e. $V_a = 4$, $V_b = 2$, $V_c = 1$, the underlying road choice is found in the active constraints above. Eg. the topmost constraint holds with equality ($V_a = V_b + 2$) but the second one does not. This means that the optimal exit road from city a is the one that leads to b. The solution is shown in Fig. 5. □

Density Function Approach The total cost depends on the density function, which in turn depends on the choices of roads. It would in principle be possible to use a naive iterative approach also for this case. There are some properties also of the density function, however, that are independent of road choice:

- The sum of the density function for the roads leading out from a city, can never be less than the production in the city plus the sum

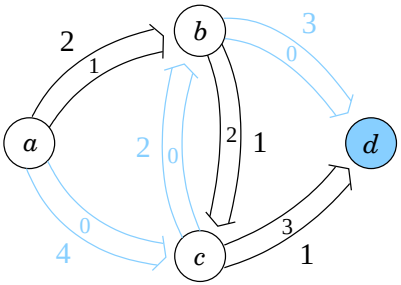


Figure 6. Density function solution of the minimum total cost example: $\lambda_{ab} = 1$, $\lambda_{ac} = 0$, $\lambda_{bc} = 2$, $\lambda_{cb} = 0$, $\lambda_{bd} = 0$, $\lambda_{cd} = 3$. The total cost is $1 \cdot 2 + 0 \cdot 4 + 2 \cdot 1 + 0 \cdot 2 + 0 \cdot 3 + 3 \cdot 1 = 7$.

of the density function for the roads leading to the city

- On a road, there can be zero or more truck transport per days. Thus, the density function can not be less than zero for any road.

The above properties make it possible to obtain the optimal flow density regardless of the corresponding road choice. The optimal value function is the smallest possible function (measured by the sum of the density function times the cost for each road) with the above properties and it can be found solving a linear program.

EXAMPLE 2—THE MINIMUM COST EXAMPLE, DENSITY FUNCTION APPROACH. Consider again the minimum cost example presented in Fig. 4. Let λ_{ab} denote the flow density from city a to b, λ_{ac} the flow from city a to c etc. The above density function properties then translate to

$$\begin{aligned} \lambda_{ab} + \lambda_{ac} &\geq 1 \\ \lambda_{bc} + \lambda_{bd} &\geq 1 + \lambda_{ab} + \lambda_{cb} \\ \lambda_{cb} + \lambda_{cd} &\geq 1 + \lambda_{ac} + \lambda_{bc} \end{aligned}$$

and

$$\lambda_{ab} \geq 0,\ \lambda_{ac} \geq 0,\ \lambda_{bc} \geq 0,\ \lambda_{bd} \geq 0,\ \lambda_{cb} \geq 0,\ \lambda_{cd} \geq 0$$

It will be shown in Sec. 3 that the *optimal* density function is the smallest function that meets the above constraints, i.e. the one minimizing $\lambda_{ab} \cdot 2 + \lambda_{ac} \cdot 4 + \lambda_{bc} \cdot 1 + \lambda_{cb} \cdot 2 + \lambda_{bd} \cdot 3 + \lambda_{cd} \cdot 1$. The solution to this linear program is shown in Fig. 6. The solution clearly shows the underlying choice of roads; the roads used are those with a flow density greater than zero. □

2.3 Extension to General Dynamical Systems

The components and principles behind the transportation problem and its solution can be extended to optimal control of dynamical systems in general. Cities correspond to states of a dynamical system and the set of all cities is the state space. The choice of various roads corresponds to dynamics that is affected by a control signal. The road tax corresponds to cost function of an optimal control problem.

The transportation problem of this section has been solved with two dual formulations for optimal control of a discrete system. A duality theory for continuous systems and hybrid systems will be presented below, the interpretation of the transportation problem of this section still being valid. First however a more general and rigorous treatment of discrete systems.

3. Duality for Discrete Systems

Define the finite sets $\bar{Q}$ and Ω. Define the functions $\nu : \bar{Q} \times \Omega \to \bar{Q}$ and $s : \bar{Q}^2 \to \mathbf{R}$. Let M denote the set of functions.

Given the control law $\mu : \bar{Q} \to \Omega$, $q_0 \in \bar{Q}$, denote the solution to the system equation

$$q(k+1) = \nu\Big(q(k), \mu\big(q(k)\big)\Big), \qquad q(0) = q_0$$

by $\phi_\mu(q_0, k)$. Define the value functions

$$V_\mu(q) = \sum_{k=0}^{\infty} s\big(\phi_\mu(q,k),\, \phi_\mu(q,k+1)\big) \qquad V^\star(q) = \inf_{\mu:\bar{Q}\to\Omega} V_\mu(q)$$

The main theorem for discrete systems can now be stated as follows:

THEOREM 1
Suppose that $\bar{Q}$ and Ω are finite sets and that Γ is a non-empty subset of $\bar{Q}$. Given $\nu : \bar{Q} \times \Omega \to \bar{Q}$ and $s : \bar{Q}^2 \to \mathbf{R}$, define ϕ_μ, and $V^\star$ as above and assume the existence of a $\mu : \bar{Q} \to \Omega$ such that $\phi_\mu(q,k) \in \Gamma$ for each $q \in \bar{Q}$ and $k \geq Q$. Define $S = \{(q,r) \in \bar{Q}^2 \,|\, r = \nu(q,\kappa),\, \kappa \in \Omega\}$. Assume that $s > 0$ on $\bar{Q}^2 \setminus \Gamma^2$ and $s = 0$ on Γ^2. Let $\psi : \bar{Q} \to \mathbf{R}$ be strictly positive. Then

$$\max_V \sum_{q\in\bar{Q}} \psi(q)V(q) = \sum_{q\in\bar{Q}} \psi(q)V^\star(q) = \min_\lambda \sum_{(q,r)\in S} s(q,r)\lambda(q,r) \qquad (1)$$

where supremum is taken over all non-negative $V : \bar{Q} \to \mathbf{R}$ such that

$$s(q,r) + V(r) - V(q) \geq 0 \qquad (q,r) \in S \tag{2}$$
$$V(q) = 0 \qquad q \in \Gamma \tag{3}$$

and the minimum is taken over the set of all non-negative $\lambda : \bar{Q}^2 \to \mathbf{R}$

$$\sum_{r|(q,r)\in S} \lambda(q,r) - \sum_{r|(r,q)\in S} \lambda(r,q) \geq \psi(q) \qquad q \in Q \setminus \Gamma \tag{4}$$

Moreover, if there exists a minimum, it is achieved by a λ' such that for each $q \in \bar{Q} \setminus \Gamma$ there is one and only one $r \in \bar{Q}$ with $\lambda'(q,r) \neq 0$. For $\mu : \bar{Q} \to \Omega$ such that $\mu(q) = \arg\max_{m\in\Omega} \lambda'\big(q, \nu(q,m)\big)$,

$$\sum_{q\in\bar{Q}} \psi(q) V_\mu(q) \leq \sum_{(q,r)\in S} s(q,r)\lambda'(q,r)$$

□

The proof is shown in Appendix 8.

Remark 4. There are several connections between Thm. 1 and Sec. 2 in addition to V and λ: The set $\bar{Q}$ corresponds to the cities, Γ is the set of cities with a dairy, S is the set of roads, $s(q,r)$ is the road tax on the road from city q to city r, $\psi(q)$ is the milk production next to the city q, and $\mu(q)$ is the choice of which road to take from city q.

4. Duality for Continuous Systems

Let $f_i \in \mathcal{C}^1(\mathbf{R}^n, \mathbf{R}^n)$, $l_i \in \mathcal{C}(\mathbf{R}^n, \mathbf{R}^m)$ with $l_i \geq 0$ for $i = 1, \ldots, M$. Let $\Gamma, X \subset \mathbf{R}^n$ be open bounded sets with C^1 boundary and $\overline{\Gamma} \subset X$. Suppose that f_i points strictly inwards on the boundary of X and the same on the boundary of Γ. Introduce $\mathcal{U}$ as the set of all non-negative $(u_1, \ldots, u_M) \in \mathcal{C}^1(X, \mathbf{R}^M)$ with $u_1(x) + \cdots + u_M(x) \equiv 1$. The solution of $\dot{x} = \sum_i u_i(x) f_i(x)$, $x(0) = x_0$ is denoted $\phi_u(x_0, t)$. Let

$$V_u(x) = \sum_i \int_0^\infty u_i\left(\phi_u(x,t)\right) l_i\left(\phi_u(x,t)\right) dt$$
$$V^*(x) = \inf_{u\in\mathcal{U}} V_u(x)$$

The main theorem for continuous systems can now be stated as follows:

THEOREM 2
Consider X, f_i, l_i, $\mathcal{U}$, ϕ_u and V^* as above. Let $l_i > 0$ outside Γ and define $\psi \in \mathcal{C}(\overline{X})$ with $\psi > 0$. Then

$$\sup_V \int_X \psi(x)V(x)dx = \int_X \psi(x)V^*(x)dx = \inf_{\rho_i} \sum_{i=1}^{M} \int_X l_i(x)\rho_i(x)dx$$

where sup is taken over non-negative $V \in \mathcal{C}^1(\mathbf{R}^n)$ such that for $i = 1, \ldots, M$

$$\nabla V(x) \cdot f_i(x) + l_i(x) > 0 \qquad x \in X \setminus \Gamma \tag{5}$$
$$V(x) = 0 \qquad x \in \Gamma \tag{6}$$

and inf is taken over $\rho_i \in \mathcal{C}_0^1(\overline{X})$ with $\rho_i > 0$ in X and

$$\sum_{i=1}^{M} \nabla \cdot (f_i(x)\rho_i(x)) > \psi(x) \qquad x \in X \setminus \Gamma \tag{7}$$

Moreover, $u := (\rho_1, \ldots, \rho_M)/(\sum_i \rho_i)$ is an element in $\mathcal{U}$ and satisfies

$$\int_X \psi(x)V_u(x)dx < \sum_{i=1}^{M} \int_X l_i(x)\rho_i(x)dx$$

□

The proof can be found in Paper B.

Note the analogy between all parts of Thm. 1 and Thm. 2. The difference is that the states and transport directions in this section are continuous. Eg. in each state x, the "road options" are a continuum of directions spanned by $f_1(x), \ldots, f_M(x)$ and the total flow density in state x is $\rho(x) = \sum_i \rho_i(x)$ in the direction of $\sum_i \rho_i(x) f_i(x)$.

EXAMPLE 3—THE DOUBLE INTEGRATOR I.
Consider the double integrator dynamics

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases} \tag{8}$$

with the state constraints $X = \{x \,|\, g(x) < 3\}$, where $g(x) = \sqrt{3}x_1^2 + 2x_1x_2 + \sqrt{3}x_2^2$, and the control signal constraint $|u| \leq 2$. The cost function to be minimized subject to this dynamics is $\int_0^{t_f} |x|^2 + u^2 dt$, where t_f is the

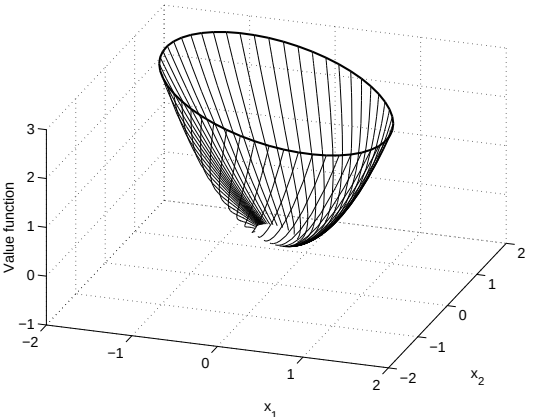


Figure 7. Value function for the double integrator example. The thick curve is the boundary of the region under consideration and the thin curves are sample trajectories starting near the boundary.

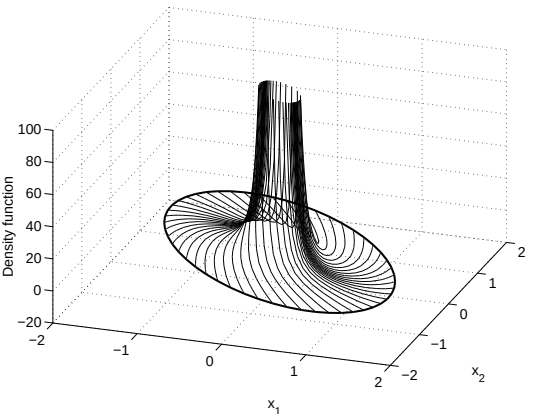


Figure 8. Density function for the double integrator example. The thick curve is the boundary of the region under consideration and the thin curves are sample trajectories starting near the boundary.

time when $\Gamma = \{x \mid g(x) < 0.1\}$ is reached. The resulting value function for this problem is $V(x) = \max(g(x) - 0.1, 0)$ and is shown in Fig. 7. For a certain choice of a positive $\psi(x)$, The density function for this example will be $\rho(x) = \frac{1}{g^2(x)} - 1$. The density function is shown in Fig. 8.

The figures also show some trajectories starting at the boundary of X.

It is clearly seen in Fig. 8 that the trajectories get denser as they approach Γ. As a consequence, the density function grows quickly towards Γ. □

Remark 5. Contrary to the conditions of Thm. 2, the set X of Ex. 3 is not invariant under all $u(x)$. For a similar example that matches the theorem without compromises, let the dynamics be $\dot{x}_1 = x_2 - \varepsilon x_1$ and $\dot{x}_2 = u - \varepsilon x_2$ for some $\varepsilon > 0$ and consider an X large enough to be invariant under the constraint $|u| \leq 2$.

5. Duality for Hybrid Systems

The hybrid systems version is presented for an autonomous system for notational convenience. The theory of this section is readily extended to systems with control signal.

Let $\bar{Q} = \{1, 2, \ldots, Q\}$, $X \subset \mathbf{R}^n$, $\nu : X \times \bar{Q} \to \bar{Q}$, and $f : X \times \bar{Q} \to \mathbf{R}^n$, with $f(x, q)$ continuously differentiable in x. A *trajectory* of the autonomous hybrid system

$$\begin{cases} \dot{x}(t) = f(x(t), q(t)) \\ q(t) = \nu(x(t), q(t^-)) \end{cases} \tag{9}$$

is defined as the collection (T, x, q) with the following properties. T is an increasing finite sequence of real numbers $0 = t_0, t_1, t_2, \ldots, t_M = \infty$, $x : [0, \infty] \to X$ is absolutely continuous, $q : [0, \infty] \to \bar{Q}$ is constant in each interval $[t_k, t_{k+1})$, $k = 0, 1, \ldots, M-1$, and

$$\begin{cases} \dot{x}(t) = f(x(t), q(t)) & \text{for almost all } t \geq 0 \\ q(t) = \nu(x(t), q(t)) = q(t_k) & t \in (t_k, t_{k+1}) \\ q(t_{k+1}) = \nu(x(t_{k+1}), q(t_k)) & k = 1, 2, \ldots, M-1 \end{cases} \tag{10}$$

Given non-negative functions $l : X \times \bar{Q} \to \mathbf{R}$, and $s : X \times \bar{Q} \times \bar{Q} \to \mathbf{R}$ with $l(x, q)$ continuous in x, s measurable, $s(x, q, r) = \infty$ for $(x, q, r) \in X \times \bar{Q} \times \bar{Q}$ such that $\nu(x, q) \neq r$, define

$$V^\star(x_0, q_0) = \inf\left\{ \int_0^\infty l\big(x(t), q(t)\big) dt + \sum_{k=1}^{M-1} s\big(x(t_k), q(t_{k-1}), q(t_k)\big) \right\} \tag{11}$$

where infimum is taken over all trajectories with $\big(x(0), q(0)\big) = (x_0, q_0)$. Then, the main theorem can be stated as follows:

THEOREM 3
Suppose that $X \subset \mathbf{R}^n$ is open and bounded with $\mathcal{C}^1$ boundary, $\bar{Q} = \{1, 2, \ldots, Q\}$, and that $\Gamma \subset X \times \bar{Q}$, where $\Gamma_q = \{x | (x,q) \in \Gamma\}$ is open with $\mathcal{C}^1$ boundary for each $q \in \bar{Q}$. Given $\bar{Q}$, f, ν, l, s, and $V^\star$ as above, assume that $f(x,q)$ points inwards at the boundary of X and at the boundary of Γ_q and that $s(\cdot,\cdot,\cdot) > 0$, and $l(x,q) > 0$ for $x \notin \Gamma_q$. Define $\psi : \overline{X} \times \bar{Q} \to \mathbf{R}$ such that $\psi(x,q)$ is continuous in x and strictly positive.

Then

$$\inf_{\rho,\lambda} \sum_{q\in\bar{Q}} \int_X \left(\rho(x,q) l(x,q) + \sum_{r\in\bar{Q}} \lambda(x,q,r) s(x,q,r) \right) dx$$
$$\leq \sup_V \sum_{q\in\bar{Q}} \int_X \psi(x,q) V(x,q) dx \leq \sum_{q\in\bar{Q}} \int_X \psi(x,q) V^\star(x,q) dx \quad (12)$$

where supremum is taken over all non-negative functions $V : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$, with $V(x,q)$ continuously differentiable in x such that

$$\nabla_x V(x,q) \cdot f(x,q) + l(x,q) > 0 \qquad (x,q) \in (X \times \bar{Q}) \setminus \Gamma \quad (13)$$
$$V(x,r) - V(x,q) + s(x,q,r) > 0 \qquad x \in X,\ q,r \in \bar{Q} \quad (14)$$
$$V(x,q) = 0 \qquad (x,q) \in \Gamma \quad (15)$$

and infimum is taken over all $\rho : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$ with $\rho(x,q)$ continuously differentiable in x, $\rho(x,q) > 0$ for $x \in X$, $\rho(x,q) = 0$ for x outside $\overline{X}$ and $\lambda : \mathbf{R}^n \times \bar{Q} \times \bar{Q} \to \mathbf{R}$ with $\lambda(x,q,r) \geq 0$ continuous in x such that $\lambda(x,q,r) = 0$ for x outside $\overline{X}$ and

$$\nabla_x \cdot \big(f(x,q)\rho(x,q)\big) + \sum_{r\in\bar{Q}} \big(\lambda(x,q,r) - \lambda(x,r,q)\big) > \psi(x,q)$$
$$(x,q) \in (X \times \bar{Q}) \setminus \Gamma \quad (16)$$

□

The proof is written in Appendix 8. Comparing Thm. 3 with Thm. 1 and Thm. 2, there is one piece missing in the hybrid case for a complete analogy to the purely discrete case and the purely continuous case. It remains to prove equality of the inequalities in (12).

6. Density Function Singularity

The examples below are purely continuous. Issues for hybrid systems that are also present in the purely continuous case are less complex to show

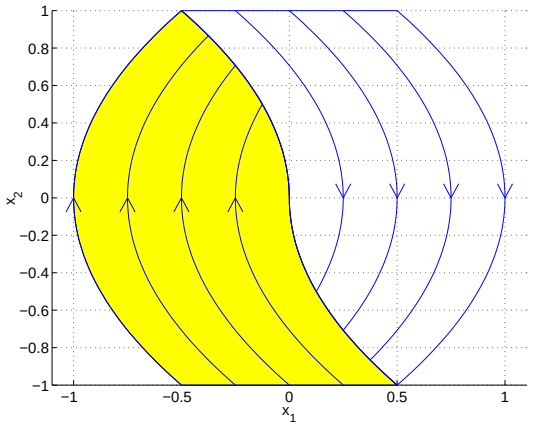


Figure 9. Optimal control law for the minimum time double integrator problem. The shaded region corresponds to $u = 1$, the white region corresponds to $u = -1$.

for the continuous case. Moreover, having overcome the problem of finding a framework with several common properties in the discrete and the continuous domain, most of the remaining difficulties are in the continuous domain.

EXAMPLE 4—THE DOUBLE INTEGRATOR II.
Consider again the double integrator dynamics given in Eq. (8). This time, however, the state space is $X = \{(x_1, x_2) | |x_1| \leq 1 - x_2^2/2, |x_2| \leq 1\}$, and the control signal is limited to $|u| \leq 1$. The optimal control problem is of minimum time, i.e. minimize $\int_0^{t_f} 1 dt$ with $x(t_f) \in \Gamma$. Here, Γ is a small ball around the origin.

The trajectories of the optimal solution are shown in Fig. 9. The control law corresponding to the optimal solution is of bang-bang type, i.e.

$$u(x_1, x_2) = \begin{cases} 1 & \text{if } (x_2 \leq 0 \text{ and } x_1 \leq x_2^2/2) \text{ or } (x_2 > 0 \text{ and } x_1 < -x_2^2/2) \\ -1 & \text{otherwise} \end{cases}$$

The region where $u = 1$ is shaded in the figure. The nature of the trajectories gives a warning for optimal control computation via the density function approach. In the optimal solution, the flow from particle production in a two-dimensional state space is collapsed to a one dimensional curve ($x_1 = -x_2^2/2$ for $x_2 > 0$, $x_1 = x_2^2/2$ for $x_2 < 0$), leading to infinite flow density along that curve.

A near optimal solution where the control signal has been smoothened around the switching curve is shown in Fig. 10. A set of particles starting

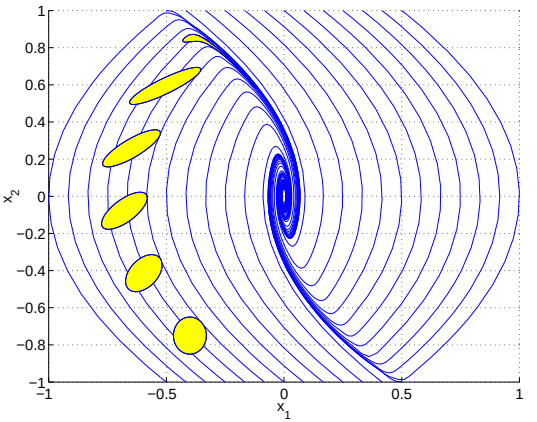


Figure 10. Evolution of a set of states along "near optimal" trajectories.

around $(-0.4, 0.75)$ is traced with a constant time interval. It is seen in the figure how the particles are squeezed when they reach the switching curve.

The intractable behavior of the density function and the control signal comes from the non-smooth control signal penalty. The cost function $l(x_1, x_2, u) = 1$ together with the control signal constraints $|u| \leq 1$ is equivalent to

$$l(x_1, x_2, u) = \begin{cases} 1 & |u| \leq 1 \\ \infty & |u| > 1 \end{cases}$$

□

7. Summary

A duality theorem for optimal control has been organized and recapitulated for dynamical systems. There is one version for discrete systems and one for continuous systems. Using a weighted sum of the value function in the whole state space as a measure of the control performance, the optimal solution can be computed with two, different, dual linear programming approaches. One of the approaches uses the value function as decision variable of the optimization, the other uses the control law as decision variable.

Having formulated the discrete system duality theorem and the continuous system version in a similar way, these theorems have been partially

merged to a hybrid system version.

An example has illustrated potential difficulties in computation of control problems with discontinuous penalty functions l,s using the dual approach.

8. Proofs of Theorems

8.1 Proof of Theorem 1 for Discrete Systems

One of the main purposes of the proof to follow is to point out analogies to the building blocks of Thm. 2, to make way for a corresponding hybrid system duality theorem. The main parts of the proof, with counterparts for the pure continuous system case, are three lemmas. Lemma 3 essentially shows equality between the maximization and the minimization of Eq. (1), call this value γ. Lemma 1 shows that the performance number of the optimal value function, call it $\gamma^\star$, is greater or equal to γ. Lemma 2 shows that $\gamma^\star \leq \gamma$ and gives the control law that achieves this optimum.

LEMMA 1
Suppose $\bar{Q} = \{1, 2, \ldots, Q\}$ and $\Gamma \subset \bar{Q}$. Given $\nu : \bar{Q} \to \bar{Q}$, $S = \{(q,r) \in \bar{Q}^2 \,|\, r = \nu(q)\}$, and $s : \bar{Q}^2 \to \mathbf{R}$, assume that $s > 0$ on $\bar{Q}^2 \setminus \Gamma^2$ and $s = 0$ on Γ^2. If $V : \bar{Q} \to \mathbf{R}$, $V(q) = 0$ for $q \in \Gamma$, and

$$s(q,r) + V(r) - V(q) \geq 0, \quad (q,r) \in S$$

then $V(q_0) \leq \sum_{k=0}^{\infty} s\big(q(k), q(k+1)\big)$ for a sequence $q(k)$ given by the dynamical system $q(k+1) = \nu(q(k))$, $q(0) = q_0 \in \bar{Q}$. □

Proof. If $\sum_{k=0}^{\infty} s\big(q(k), q(k+1)\big) < \infty$ then there exists a $\bar{k}$ such that $s\big(q(k), q(k+1)\big) = 0$ for $k \geq \bar{k}$. It follows that $q(k) \in \Gamma$ for $k \geq \bar{k}$.

$$\begin{aligned} V(q_0) &= V\big(q(0)\big) - V\big(q(\bar{k})\big) = \sum_{k=0}^{\bar{k}-1} V\big(q(k)\big) - V\big(q(k+1)\big) \\ &\leq \sum_{k=0}^{\bar{k}-1} s\big(q(k), q(k+1)\big) = \sum_{k=0}^{\infty} s\big(q(k), q(k+1)\big) \end{aligned}$$

□

LEMMA 2
Let $\bar{Q} = \{1, 2, \ldots, Q\}$ and $\Gamma \subset \bar{Q}$. Given $\nu : \bar{Q} \to \bar{Q}$ and a non-negative $s : \bar{Q} \to \mathbf{R}$, denote the solution of $q(k+1) = \nu\big(q(k)\big)$, $q(0) = q_0$ by $\phi(q_0, k)$ for $k \geq 0$. For $k < 0$, let $\phi(q_0, k) = \{q \in \bar{Q} \,|\, \phi(q, -k) = q_0\}$. Also define $V(q) = \sum_{k=0}^{\infty} s\big(\phi(q,k)\big)$. Suppose that $\lambda : \bar{Q} \to \mathbf{R}$ and $\psi : \bar{Q} \to \mathbf{R}$ are non-negative and

$$\lambda(q) - \sum_{r \in \phi(q,-1)} \lambda(r) \geq \psi(q) \quad \forall q \in \bar{Q} \setminus \Gamma$$

Then

$$\sum_{q \in \bar{Q} \setminus \Gamma} \psi(q) V(q) \leq \sum_{q \in \bar{Q} \setminus \Gamma} s(q) \lambda(q)$$

□

The proof of Lemma 2 uses the following proposition (discrete version of Liouville's theorem):

PROPOSITION 1
Let $\bar{Q} = \{1, 2, \ldots, Q\}$ and denote the set of integers by $\mathbf{Z}$. Assume that $\phi : \bar{Q} \times \mathbf{Z} \to 2^{\bar{Q}}$ has the property $\phi(q_0, k) = \{q \in \bar{Q} \,|\, \phi(q, -k) = q_0\}$ for $k < 0$. Then for $D \subset \bar{Q}$

$$\sum_{r \in D} \lambda(r) - \sum_{q \in \phi(D,-k)} \lambda(q) = \sum_{i=0}^{k-1} \sum_{q \in \phi(D,-i)} \Big(\lambda(q) - \sum_{r \in \phi(q,-1)} \lambda(r) \Big)$$

□

The proof of the proposition is in realizing that the above equation is a telescopic sum.

Remark 6. To interpret the proposition in terms of the transportation problem in Sec. 2, let D be a set of cities. Then, all transports leaving the cities in the set $\phi(D, -k)$ will after passing $k-1$ cities end up in the cities in D. The proposition states that the difference between the amount of milk transported out from D and the amount transported out from $\phi(D, -k)$ is the milk production next to the cities on the way from $\phi(D, -k)$ to D.

Proof of Lemma 2. For $q, r \in \bar{Q}$, define

$$\chi_q(r) = \begin{cases} 1, & r = q \\ 0, & r \neq q \end{cases}$$

Then

$$\begin{aligned}
&\sum_{q\in\bar{Q}} s(q)\lambda(q) \geq \sum_{q\in\bar{Q}} s(q)\{\lambda(q) - \sum_{r\in\phi(q,-N)} \lambda(r)\} \\
&= \sum_{q\in\bar{Q}} s(q) \sum_{i=0}^{N-1} \sum_{p\in\phi(q,-i)} \{\lambda(p) - \sum_{r\in\phi(p,-1)} \lambda(r)\} \geq \sum_{q\in\bar{Q}} s(q) \sum_{i=0}^{N-1} \sum_{p\in\phi(q,-i)} \psi(p) \\
&= \sum_{q\in\bar{Q}} s(q) \sum_{i=0}^{N-1} \sum_{p\in\bar{Q}} \chi_q(\phi(p,i))\psi(p) = \sum_{p\in\bar{Q}} \psi(p) \sum_{i=0}^{N-1} \sum_{q\in\bar{Q}} \chi_q(\phi(p,i))s(q) \\
&= \sum_{p\in\bar{Q}} \psi(p) \sum_{i=0}^{N-1} s(\phi(p,i))
\end{aligned}$$

The proof is complete as $N \to \infty$. □

LEMMA 3
Let Γ be a non-empty subset of $\bar{Q} = \{1, 2, \ldots, Q\}$, and $S \subset \bar{Q}^2$. Assume that $s : S \to \mathbf{R}$, $s \geq 0$, $\psi : \bar{Q} \to \mathbf{R}$, $\psi > 0$. Then

$$\max_V \sum_{q\in\bar{Q}} \psi(q)V(q) = \min_\lambda \sum_{(q,r)\in S} s(q,r)\lambda(q,r) \tag{17}$$

where maximum is taken over all non-negative $V : \bar{Q} \to \mathbf{R}$ satisfying

$$V(r) - V(q) + s(q,r) \geq 0 \qquad (q,r) \in S \tag{18}$$

$$V(q) = 0 \qquad q \in \Gamma \tag{19}$$

and minimum is taken over all $\lambda : \bar{Q}^2 \to \mathbf{R}$ with $\lambda \geq 0$ and

$$\sum_{r|(q,r)\in S} \lambda(q,r) - \sum_{k|(k,q)\in S} \lambda(k,q) > \ \psi(q) \qquad q \in \bar{Q} \setminus \Gamma \tag{20}$$

□

Proof. Define the following two subsets of $K = \mathbf{R}^{Q+1}$

$$\begin{aligned}
K_1 = \Big\{ \Big(&- \sum_{(q,r)\in S} s(q,r)\lambda(q,r) + \gamma, \sum_{r|(1,r)\in S} \lambda(1,r) - \sum_{k|(k,1)\in S} \lambda(k,1) - \psi(1), \\
&\sum_{r|(2,r)\in S} \lambda(2,r) - \sum_{k|(k,2)\in S} \lambda(k,2) - \psi(2), \ldots, \\
&\sum_{r|(Q,r)\in S} \lambda(Q,r) - \sum_{k|(k,Q)\in S} \lambda(k,Q) - \psi(Q) \Big) \Big| \ \lambda : \bar{Q}^2 \to \mathbf{R},\ \lambda \geq 0 \Big\}
\end{aligned}$$

$$K_2 = \{(z, h_1, \ldots, h_Q) \in K \mid z \geq 0,\ h_q \geq 0 \text{ for } q \in \bar{Q} \setminus \Gamma\}$$

We will prove the following five statements to be equivalent:

I The number γ is not larger than the right hand side of (17).

II K_1 contains no interior point of K_2.

III There exists $k^\star \in K^\star, k^\star \neq 0$ such that

$$\sup_{k_1 \in K_1} \langle k^\star, k_1 \rangle \leq \inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle, \tag{21}$$

where $K^\star$ is the dual space of K, i.e. $K^\star = \mathbf{R}^{Q+1}$

IV There exists a nonzero $(Q+1)$-tuple $(a, \phi_1, \ldots, \phi_Q) \in \mathbf{R}^{Q+1}$ with all elements non-negative, $\phi_q = 0$ for $q \in \Gamma$, $\sum_{q \in \bar{Q}} \phi_q \psi(q) \geq a\gamma$, and

$$\phi_r - \phi_q + as(q,r) \geq 0 \qquad\qquad (q,r) \in S \tag{22}$$

V The number γ is not larger than the left hand side of (17).

The equivalence **I**⇔**II** is trivial once it is noted that $(z, h_1, h_2, \ldots, h_Q) \in K_2$ is an interior point if and only if $z > 0$ and $h_1, \ldots, h_Q > 0$. The second equivalence **II**⇔**III** holds because of the following separation property of convex sets [Luenberger, 1969; Rudin, 1991]:

Let K be a normed vector space and denote its dual $K^\star$. Let K_1 and K_2 be convex sets in K such that K_2 has interior points and K_1 contains no interior point of K_2. Then there is a closed hyperplane separating K_1 and K_2; i.e., there is a $k^\star \in K^\star$, $k^\star \neq 0$ such that

$$\sup_{k_1 \in K_1} \langle k^\star, k_1 \rangle \leq \inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle, \tag{23}$$

Note that the current application of this property is a simple special case where K is of finite dimension. The general formulation above, however, will also be applied to the more complex counterparts of continuous and hybrid systems.

To show that **III**⇔**IV**, let $k^\star = (a, \phi_1, \ldots, \phi_Q) \in K^\star = \mathbf{R}^{Q+1}$. Expand the right hand side of (21) to

$$\inf_{k_2 \in K_2} \langle k^\star, k_2 \rangle = \min_{(z, h_1, \ldots, h_Q) \in K_2} \Big\{ az + \sum_{q=1}^{Q} \phi_q h_q \Big\} \tag{24}$$

The right hand side is equal to zero if and only if a and $\phi_1, \ldots, \phi_Q$ are non-negative and $\phi_q = 0$ for $q \in \Gamma$. Otherwise it is $-\infty$.

The left hand side of (21) can be expanded to

$$\begin{aligned}\sup_{k_1\in K_1}\langle k^\star,k_1\rangle &= \max_{\lambda}\Big\{a\Big(-\sum_{(q,r)\in S}s(q,r)\lambda(q,r)+\gamma\Big)\\ &\qquad+\sum_{q\in\bar{Q}}\phi_q\Big(\sum_{r|(q,r)\in S}\lambda(q,r)-\sum_{k|(k,q)\in S}\lambda(k,q)-\psi(q)\Big)\Big\}\\ &= \max_{\lambda}\sum_{(q,r)\in S}\lambda(q,r)\Big\{\phi_q-\phi_r-as(q,r)\Big\}+a\gamma-\sum_{q\in\bar{Q}}\phi_q\psi(q)\end{aligned}$$

The maximum is taken over $\lambda : \bar{Q}^2 \to \mathbf{R}$, $\lambda \geq 0$ and the value is equal to $a\gamma - \sum_{q\in\bar{Q}}\phi_q\psi(q)$ if and only if (22) holds, otherwise it is $+\infty$. The statement **III** is thus equivalent to **IV**.

The implication from **V** to **IV** is obtained with $a = 1$ and $\phi_q = V(q)$ where V is the function that maximizes Eq. (17). For the opposite implication, it can be seen that the existence of a non-zero $(Q+1)$-tuple $(a,\phi_1,\ldots,\phi_Q)$ as in **IV**, also implies the existence of a $(Q+1)$-tuple with a strictly positive. For such a $(Q+1)$-tuple, define $V(q) = \phi_q/a$. This $V : \bar{Q} \to \mathbf{R}$ will satisfy (18)–(19) and $\sum_{q\in\bar{Q}}\psi(q)V(q) \geq \gamma$. □

Proof of Theorem 1 Lemma 3 proves equality between the maximum and the minimum in (1). The next few lines will show the existence of a mono flow solution, i.e. a solution λ such that for each $q \in \bar{Q} \setminus \Gamma$ there is one and only one $r \in \bar{Q}$ with $\lambda(q,r) \neq 0$, whenever the optimization problem of Lemma 3 has a finite solution.

Note that any $\lambda : \bar{Q}^2 \to \mathbf{R}$ that satisfies $\lambda \geq 0$ and (20) must have the property that for each $q \in \bar{Q} \setminus \Gamma$ there exists *at least* one $r \in \bar{Q}$ with $\lambda(q,r) > 0$. To show the existence of a solution with *no more* than one $r \in \bar{Q}$ with $\lambda(q,r) > 0$ for each $q \in \bar{Q} \setminus \Gamma$, recall that if a linear program with n decision variables has a finite solution, then there exists a solution with n active constraints. The minimization part of Lemma 3 contains Q^2 decision variables, i.e. $\lambda(q,r)$ with $(q,r) \in \bar{Q}^2$. Each of these variables is constrained to be non-negative and there are $\#(\bar{Q} \setminus \Gamma) < Q^2$ additional constraints.[3] Thus, in a solution with Q^2 active constraints, $\lambda(q,r)$ is non-zero for no more than $\#(\bar{Q} \setminus \Gamma)$ combinations of q and r. This solution is a mono flow solution.

Denote the optimal value of (1) $\gamma^\star$ and the corresponding decision variables V' and λ'. The control $\mu(q) = \arg\max_{m\in\Omega_\mu}\lambda'(q,\nu(q,m))$ gives an autonomous system with an S for which λ' is feasible in (4) and, according to Lemma 2, $\sum_{q\in\bar{Q}}\psi(q)V_\mu(q) \leq \sum_{(q,r)\in S}s(q,r)\lambda'(q,r)$. Thus, $\sum_{q\in\bar{Q}}\psi(q)V^\star(q) \leq \gamma^\star$.

[3]The symbol # denotes an operator that gives the cardinality of a set.

By Lemma 1, $V' \leq V^\star$ and thus $\gamma^\star \leq \sum_{q\in\bar{Q}} \psi(q)V^\star(q)$ and the proof is complete. □

8.2 Proof of Theorem 3 for Hybrid Systems

The proof consists of two different lemmata that have their counterparts for purely discrete systems. Lemma 4 corresponds to Lemma 1, and Lemma 5 corresponds to Lemma 3.

LEMMA 4
Suppose that $X \subset \mathbf{R}^n$ is open and bounded, $\bar{Q} = \{1, 2 \ldots, Q\}$ and that $\Gamma \subset X \times \bar{Q}$, where $\Gamma_q = \{x|(x,q) \in \Gamma\}$ is open for all $q \in \bar{Q}$. Given $f : X \times \bar{Q} \to \mathbf{R}^n$, $f(x,q)$ continuous in x, $l : X \times \bar{Q} \to \mathbf{R}$, $l(x,q)$ continuous in x, and $s : X \times \bar{Q} \times \bar{Q} \to \mathbf{R}$, assume that $l(x,q) \geq 0$ with strict inequality for $x \notin \Gamma_q$, and that $s(x,q,r) \geq 0$.

If V is a map from $X \times \bar{Q}$ to $\mathbf{R}$, with $V(x,q)$ continuously differentiable in x and

$$\begin{aligned} \nabla_x V(x,q) \cdot f(x,q) + l(x,q) &\geq 0 && (x,q) \in (X \times \bar{Q}) \setminus \Gamma \\ V(x,r) - V(x,q) + s(x,q,r) &\geq 0 && x \in X,\ q,r \in \bar{Q} \\ V(x,q) &= 0 && (x,q) \in \Gamma \end{aligned}$$

then $V(x_0,q_0) \leq \int_0^\infty l\big(x(t),q(t)\big)dt + \sum_{k=1}^{M-1} s\big(x(t_k),q(t_{k-1}),q(t_k)\big)$ where $\big(x(t),q(t)\big)$ with switching times t_0, $t_1, \ldots$, t_{M-1} is a trajectory of (9) with $\big(x(0),q(0)\big) = (x_0,q_0)$. □

Proof. To simplify notation, let x_k and q_k denote $x(t_k)$ and $q(t_k)$ respectively. If $\int_0^\infty l\big(x(t),q(t)\big)dt + \sum_{k=1}^{M-1} s\big(x_k,q_{k-1},q_k\big) < \infty$, then there exists a sequence $\tau_1 < \tau_2 < \tau_3 < \ldots$ with $\tau_1 > t_{M-1}$ such that $\lim_{i\to\infty} \tau_i = \infty$ and

$\lim_{i\to\infty} l\big(x(\tau_i), q(\tau_i)\big) = 0$. It follows that $x(\tau_i) \to \Gamma_{q(\tau_i)}$ as $i \to \infty$ and

$$\begin{aligned}
&V(x_0, q_0) - V\big(x(\tau_i), q(\tau_i)\big) \\
&= \sum_{k=1}^{M-1} \big(V(x_{k-1}, q_{k-1}) - V(x_k, q_{k-1}) + V(x_k, q_{k-1}) - V(x_k, q_k)\big) \\
&\quad + V(x_{M-1}, q_{M-1}) - V\big(x(\tau_i), q(\tau_i)\big) \\
&= \sum_{k=1}^{M-1} \Big\{ - \int_{t_{k-1}}^{t_k} \frac{d}{dt} V\big(x(t), q_{k-1}\big) dt + V(x_k, q_{k-1}) - V(x_k, q_k) \Big\} \\
&\quad - \int_{t_{M-1}}^{\tau_i} \frac{d}{dt} V\big(x(t), q_{M-1}\big) dt \\
&\le \sum_{k=1}^{M-1} \Big\{ \int_{t_{k-1}}^{t_k} l\big(x(t), q_{k-1}\big) dt + s(x_k, q_{k-1}, q_k) \Big\} \\
&\quad + \int_{t_{M-1}}^{\tau_i} l\big(x(t), q_{M-1}\big) dt
\end{aligned}$$

In the limit as $i \to \infty$ we get the desired inequality. □

LEMMA 5
Suppose that X is open and bounded with $\mathcal{C}^1$ boundary and that $\bar{Q} = \{1, 2, \ldots, Q\}$, $\Gamma \subset X \times \bar{Q}$, where $\Gamma_q = \{x | (x, q) \in \Gamma\}$ is open and bounded with $\mathcal{C}^1$ boundary for all $q \in \bar{Q}$. Let $f : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}^n$, $f(x, q)$ continuously differentiable in x with $f(x, q)$ pointing inwards at the boundary of X and at the boundary of Γ_q, $\nu : \mathbf{R}^n \times \bar{Q} \to \bar{Q}$, $l : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$, $l(x, q) \ge 0$ continuous in x with strict inequality for (x, q) outside Γ, $s : \mathbf{R}^n \times \bar{Q} \times \bar{Q} \to \mathbf{R}$, s measurable, $s > 0$ with $s(x, q, r) = \infty$ for (x, q, r) such that $\nu(x, q) \ne r$, and $\psi : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$ such that $\psi(x, q)$ is continuous in x and strictly positive. Then

$$\begin{aligned}
&\sup_V \sum_{q\in\bar{Q}} \int_X \psi(x, q) V(x, q) dx \\
&\qquad \ge \inf_{\rho, \lambda} \sum_{q\in\bar{Q}} \int_X \Big(\rho(x, q) l(x, q) + \sum_{r\in\bar{Q}} \lambda(x, q, r) s(x, q, r) \Big) dx \qquad (25)
\end{aligned}$$

where supremum is taken over all non-negative functions $V : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$, with $V(x, q)$ continuously differentiable in x such that

$$\nabla_x V(x, q) \cdot f(x, q) + l(x, q) > 0 \qquad (x, q) \in (X \times \bar{Q}) \setminus \Gamma \qquad (26)$$

$$V(x, r) - V(x, q) + s(x, q, r) > 0 \qquad x \in X,\ q, r \in \bar{Q} \qquad (27)$$

$$V(x, q) = 0 \qquad (x, q) \in \Gamma \qquad (28)$$

and infimum is taken over all $\rho : \mathbf{R}^n \times \bar{Q} \to \mathbf{R}$ with $\rho(x,q)$ continuously differentiable in x, $\rho(x,q) > 0$ for $x \in X$, $\rho(x,q) = 0$ for x outside $\overline{X}$ and $\lambda : \mathbf{R}^n \times \bar{Q} \times \bar{Q} \to \mathbf{R}$ with $\lambda(x,q,r) \geq 0$ continuous in x, and $\lambda(x,q,r) = 0$ for x outside X such that

$$\nabla_x \cdot \big(f(x,q)\rho(x,q)\big) + \sum_{r\in\bar{Q}} \big(\lambda(x,q,r) - \lambda(x,r,q)\big) > \psi(x,q) \qquad (x,q) \in (X \times \bar{Q}) \setminus \Gamma \quad (29)$$

□

Proof. To simplify notation, let $f_q(x)$, $l_q(x)$, $s_{q,r}(x)$, $\psi_q(x)$, $\rho_q(x)$ and $\lambda_{q,r}(x)$ denote $f(x,q)$, $l(x,q)$, $s(x,q,r)$, $\psi(x,q)$, $\rho(x,q)$, and $\lambda(x,q,r)$ respectively. Define the following two subsets of $K = \mathbf{R} \times C(\overline{X})^Q$.

$$\begin{aligned} K_1 = \Big\{ & -\sum_{q\in\bar{Q}} \int_X (\rho_q l_q + \sum_{r\in\bar{Q}} \lambda_{q,r} s_{q,r}) dx + \gamma, \\ & \nabla \cdot (f_1\rho_1) + \sum_{r\in\bar{Q}} (\lambda_{1,r} - \lambda_{r,1}) - \psi_1, \\ & \nabla \cdot (f_2\rho_2) + \sum_{r\in\bar{Q}} (\lambda_{2,r} - \lambda_{r,2}) - \psi_2, \\ & \vdots \\ & \nabla \cdot (f_Q\rho_Q) + \sum_{r\in\bar{Q}} (\lambda_{Q,r} - \lambda_{r,Q}) - \psi_Q \\ & \mid \rho_q \in C_0^1(\overline{X}), \rho_q > 0 \text{ in } X,\ \lambda_{r,q} \in C_0(X) \geq 0 \Big\} \\ K_2 = \Big\{ & (z, h_1, h_2, \ldots, h_Q) \mid z \geq 0,\ h_q(x) > 0,\ x \in X \setminus \Gamma_q,\ q \in \bar{Q} \Big\} \end{aligned}$$

We will prove the implications **I**⇔**II**⇔**III** ⇔**IV**⇒**V**.

I The number γ is not larger than the right hand side of (25).

II K_1 contains no interior point of K_2.

III There exists $k^\star \in K^\star, k^\star \neq 0$ such that

$$\sup_{k_1\in K_1} \langle k^\star, k_1 \rangle \leq \inf_{k_2\in K_2} \langle k^\star, k_2 \rangle, \qquad (30)$$

where $K^\star$ is the dual space of K, i.e. $K^\star = \mathbf{R} \times \big(C(\overline{X})^\star\big)^Q$

IV There exists a nonzero $(Q+1)$-tuple $(a, \phi_1, \phi_2, \ldots, \phi_Q)$ where $a \geq 0$ is a number and $\phi_q \geq 0$, $q \in \bar{Q}$ are measures of bounded variation on $\overline{X}$, vanishing inside Γ_q, such that $\sum_{q\in\bar{Q}} \langle \phi_q, \psi_q \rangle := \sum_{q\in\bar{Q}} \int_X \psi_q(x) d\phi_q(x) \geq a\gamma$ and

$$al_q + \nabla\phi_q \cdot f_q \geq 0 \qquad \text{in } X \text{ for } q \in \bar{Q} \tag{31}$$

$$as_{q,r} - \phi_q + \phi_r \geq 0 \qquad \text{in } X \text{ for } q, r \in \bar{Q} \tag{32}$$

The derivative $\nabla\phi_q$ is interpreted in the sense of distributions.

V The number γ is not larger than the left hand side of (25).

The equivalence **I**⇔**II** is trivial once it is noted that $(z, h_1, \ldots, h_Q) \in K_2$ is an interior point if and only if $z > 0$ and $h_q(x) > 0$, $q \in \bar{Q}$ for $x \in \overline{X}$. The second equivalence **II**⇔**III** holds because of the separation property of convex sets quoted on page 67.

To show that **III**⇔**IV**, let $k^\star = (a, \phi_1, \ldots, \phi_Q) \in K^\star = \mathbf{R} \times \left(\mathcal{C}(\overline{X})^\star\right)^Q$. The space $\mathcal{C}(\overline{X})^\star$ is the set of measures of bounded variation and support in X [Dunford and Schwartz, 1958] . Expand the right hand side of (30) to

$$\inf_{k_2\in K_2} \langle k^\star, k_2 \rangle = \inf_{(z,h_1,\ldots,h_Q)\in K_2} \left\{ za + \sum_{q\in\bar{Q}} \langle \phi_q, h_q \rangle \right\} \tag{33}$$

The right hand side is equal to zero if and only if a and $\phi_1, \ldots, \phi_Q$ are non-negative and $\phi_q = 0$ in Γ_q, $q \in \bar{Q}$. Otherwise it is $-\infty$.

The left hand side of (30) can be expanded to

$$\begin{aligned}
\sup_{k_1\in K_1} \langle k^\star, k_1 \rangle = \sup_{\rho_q,\, \lambda_{r,q}} \Bigg\{ & a\Big(-\sum_{q\in\bar{Q}} \int_X (\rho_q l_q + \sum_{r\in\bar{Q}} \lambda_{q,r} s_{q,r}) dx + \gamma \Big) \\
& + \sum_{q\in\bar{Q}} \Big\langle \phi_q, \nabla\cdot(f_q\rho_q) + \sum_{r\in\bar{Q}} (\lambda_{q,r} - \lambda_{r,q}) - \psi_q \Big\rangle \Bigg\} \\
= & \sum_{q\in\bar{Q}} \sup_{\rho_q} \Big\langle -al_q - \nabla\phi_q \cdot f_q, \rho_q \Big\rangle \\
& + \sum_{q\in\bar{Q}} \sum_{r\in\bar{Q}} \sup_{\lambda_{q,r}} \Big\langle -as_{q,r} + \phi_q - \phi_r, \lambda_{q,r} \Big\rangle + a\gamma - \sum_{q\in\bar{Q}} \Big\langle \phi_q, \psi_q \Big\rangle
\end{aligned}$$

The supremum is taken over $\rho_q \in \mathcal{C}^1(\mathbf{R}^n)$ with support in X, $\rho_q > 0$ in X and $\lambda_{r,q} \in \mathcal{C}(\mathbf{R}^n) \geq 0$ with support in X. The value is equal to $a\gamma - \sum_{q\in\bar{Q}} \langle \phi_q, \psi_q \rangle$ if and only if (31) and (32) holds, otherwise it is $+\infty$. The statement **III** is thus equivalent to **IV**.

It remains to prove the implication from **IV** to **V**. Hence, assume existence of a non-zero $(Q+1)$-tuple $(a, \phi_1, \ldots, \phi_Q)$ as in **IV**. Note that ϕ_q

$q \in \bar{Q}$ can be identified with distributions of order zero on $\mathbf{R}^n$, which are identically zero outside $\overline{X}$. If $a = 0$, define $\bar{a} = \bar{\varepsilon}$ and $\bar{\gamma} = \gamma$. If $a > 0$, define $\bar{a} = a(1+\bar{\varepsilon})$ and $\bar{\gamma} = \gamma/(1+2\bar{\varepsilon})$. In both cases, for sufficiently small $\bar{\varepsilon} > 0$, it holds that $\sum_{q\in\bar{Q}} \langle \phi_q, \psi_q \rangle > \bar{a}\bar{\gamma}$ and

$$\bar{a} l_q + \nabla \phi_q \cdot f_q > 0 \qquad \text{in } X \setminus \Gamma_q \text{ for } q \in \bar{Q} \tag{34}$$

$$\bar{a} s_{q,r} - \phi_q + \phi_r > 0 \qquad \text{in } X \text{ for } q, r \in \bar{Q} \tag{35}$$

Here, the assumption that all f_q points strictly inwards on the boundary of X is critical to guarantee that $\nabla_q \phi \cdot f_q \geq 0$ on the boundary of X.

Note that ϕ_q cannot be singular on the boundary of Γ_q, due to (34) in combination with the assumption that f_q points strictly inwards on the boundary of Γ_q. Hence, one can write $\phi_q = \sum_{i=1}^{\infty} \phi_{q,i}$ where each $\phi_{q,i}$ is zero within a distance ε^i from Γ_q. Each of the terms can be regularized by convolution with a smooth non-negative function having support in a ball of sufficiently small radius and integral equal to one. In this way, ϕ_q can be approximated by a smooth function function $\phi_{q,0}$, vanishing inside Γ_q, satisfying $\sum_{q\in\bar{Q}} \langle \phi_{q,0}, \psi_q \rangle > \bar{a}\bar{\gamma}$ and

$$0 < \bar{a} l_q + \nabla \phi_{q,0} \cdot f_q \quad \text{ in } \mathbf{R}^n \setminus \Gamma_q, \ q \in \bar{Q}$$

Finally, $V(x,q) = \phi_{q,0}/\bar{a} \in \mathcal{C}^1(\mathbf{R}^n)$ gives (6)-(7) and

$$\sum_{q\in\bar{Q}} \int_X \psi(x,q) V(x,q) dx > \bar{\gamma}$$

Condition **V** follows and the proof is complete. □

Paper D

Hybrid Control Laws From Convex Dynamic Programming

Sven Hedlund and Anders Rantzer

Abstract

In a previous paper, we showed how classical ideas for dynamic programming in discrete networks can be adapted to hybrid systems. The approach is based on discretization of the continuous Bellman inequality which gives a lower bound on the optimal cost. The lower bound is maximized by linear programming to get an approximation of the optimal solution.

In this paper, we apply ideas from infinite-dimensional convex analysis to get an inequality which is dual to the well known Bellman inequality. The result is a linear programming problem that gives an estimate of the approximation error in the previous numerical approaches.

Keywords optimal control, duality, convex dynamic programming, hybrid systems.

1. Introduction

One of the most important aspects of the current research activity in the field of hybrid systems is the exchange of ideas between the research fields of discrete and continuous dynamics. This paper can be viewed as an attempt to approach optimal continuous and hybrid systems using a classical linear programming perspective for discrete transportation and flow problems.

The transportation problem was formulated by Hitchcock [Hitchcock, 1941] and a classic reference for network flow theory is [Ford and Fulkerson, 1962]. A continuous analog is the "Monge-Kantorivich mass transfer problem" dating back to Monge in 1781 and nicely surveyed in [Evans, 1997].

A primal/dual formulation of a continuous optimal control problem for a continuous system is presented in [Vinter, 1993] based on the Bellman inequality. A central concept is L.C. Youngs notion of generalized flow [Young, 1969].

Discretization of the Bellman inequality for numerical computations can be done in several ways [Branicky and Mitter, 1995; Rantzer and Johansson, 2000; Rantzer, 1999; Hedlund and Rantzer, 1999].

This paper is devoted to an inequality which is dual to the "Hybrid Bellman inequality" which served as basis for the computations in [Hedlund and Rantzer, 1999]. The dual gives valuable information about the conservatism introduced by the discretization.

In Section 2, a discrete transportation problem is discussed as a preparation for the hybrid problem of Section 3.

2. Discrete Problem Formulation

Define a discrete dynamic system as

$$q(k+1) = v(q(k), \mu(k)) \tag{1}$$

where $q \in Q = \{1, 2, \ldots, N\}$ is the discrete state, $\mu \in \Omega_\mu$ is the input signal of the system, and $v : Q \times \Omega_\mu \to Q$ is a function telling what state transitions are possible.

Let $\Gamma \subset Q$ be the set of final states and consider the optimal control problem of bringing the system from an initial state, $q_0 \in Q$, to a final state, $q_f \in \Gamma$, while minimizing

$$V_\mu(q_0) = \sum_{k=1}^{k_f} s(q(k-1), q(k)) \tag{2}$$

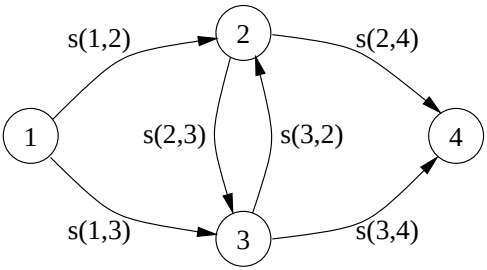


Figure 1. A simple discrete dynamic system.

Here $s(q,r) > 0$, $(q,r) \in S$ is the cost for switching from state q to r. The set S contains all pairs (q,r) such that a transition from mode q to mode r is possible. The time when Γ is reached is represented by the variable k_f.

The function V is commonly referred to as the value function or "cost-to-go" function of the system.

EXAMPLE 1—THE TRANSPORTATION PROBLEM
A simple discrete dynamic system is shown in Fig. 1. Here the final state is $\Gamma = \{4\}$ and the goal is to find the cheapest path from the initial state $q_0 = 1$. □

Define the optimal value function $V^\star(q_0) = \min_{\mu\in\Omega_\mu} V_{\mu(\cdot)}(q_0)$. A lower bound on $V^\star$ is then given by any function $V : Q \to \mathbf{R}^+$ that satisfies

$$0 \le s(q,r) + V(r) - V(q) \qquad (q,r) \in S \tag{3}$$
$$0 = V(q) \qquad q \in \Gamma \tag{4}$$

Moreover, a bound on $V^\star$ can be found for all possible initial states simultaneously by solving *one* LP, maximizing a sum of V for those states, i.e.

$$\max_{V(q)} \sum_{q\in Q\setminus\Gamma} \psi(q)V(q), \tag{5}$$

where a suitable choice of ψ would be $\psi(q) \equiv 1$.

EXAMPLE 2—THE LP APPROACH TO THE TRANSPORTATION PROBLEM
The transportation problem of Fig. 1 can be viewed as an LP problem according to (3)–(5), i.e.

$$\begin{aligned} \text{maximize} \quad & V(1) + V(2) + V(3) \\ \text{subject to} \quad & V(1) - V(2) \leq s(1,2) \\ & V(1) - V(3) \leq s(1,3) \\ & \vdots \\ & V(3) - V(4) \leq s(3,4) \\ & V(4) = 0 \end{aligned}$$

A common way to solve an LP of this structure is Dijkstra's algorithm [Cormen *et al.*, 1989]. □

2.1 Upper Bound on the Value Function via the Dual Problem

Knowing that every solution to the above problem gives a lower bound on the value function, it would be interesting to compute an upper bound as well. If the gap between the lower and upper bound is small, then the bounds are close to the optimal function. Fortunately, there is a dual LP problem that gives an upper bound on the value function.

EXAMPLE 3—DUAL LP OF THE TRANSPORTATION PROBLEM
The dual LP of the transportation problem is to

$$\begin{aligned} \text{minimize} \quad & s(1,2)\lambda_{12} + s(1,3)\lambda_{13} + s(2,3)\lambda_{23} \\ & + s(3,2)\lambda_{32} + s(2,4)\lambda_{24} + s(3,4)\lambda_{34} \\ \text{subject to} \quad & -\lambda_{12} + \lambda_{23} - \lambda_{32} + \lambda_{24} \geq 1 \\ & -\lambda_{13} - \lambda_{23} + \lambda_{32} + \lambda_{34} \geq 1 \\ & \lambda_{12} + \lambda_{13} \geq 1 \end{aligned}$$

where $\lambda_{qr} \geq 0$ are the decision variables.

□

In general, every constraint in a primal problem appears as a variable in the dual problem. For the transportation problem, every possible transition gives rise to a constraint via the switching cost, s, and the corresponding dual variable when switching from node q to r is denoted λ_{qr}. (Conversely, every variable in the primal problem gives rise to a constraint in the dual problem, so $V(1)$, $V(2)$, and $V(3)$ correspond to the λ inequalities above.)

An interpretation of the dual problem can be given in terms of mass flow instead of the single mass unit transportation of the primal problem. The variable λ_{qr} is the flow from node q to r. There is a unit mass production in the starting states, and mass consumption in the end states. The dual problem is then to minimize the cost of the overall flow for this system. Conservation of mass implies that the production in a single node

must not exceed the net flow out from the node. This corresponds to the inequality constraints for λ.

The general formulation of the dual problem to the maximization of (5) subject to (3) and (4) is thus

$$\min_{\lambda_{qr}} \sum_{(q,r)\in S} \lambda_{qr} s(q,r) \tag{6}$$

$$\text{subject to} \sum_{r|(q,r)\in S} \lambda_{qr} - \sum_{r|(r,q)\in S} \lambda_{rq} \geq \psi(q),\ \forall q \in Q\backslash\Gamma \tag{7}$$

The trajectories that solve the original optimal control problem of minimizing (2) subject to the dynamics in (1) are easily found in the solutions to the primal and dual problem above. In the solution to the primal problem, the constraint (3) is active (equality holds) only for transitions (q,r) along the optimal trajectory. The same information is available in the variables of the dual problem: if there is a unique solution to the problem, then λ_{qr} is greater than zero for transitions (q,r) along the optimal trajectory and zero elsewhere.

3. Hybrid Problem Formulation

The idea of how to obtain a lower bound on the value function of the discrete dynamic system was the primal problem of maximizing the sum of the value function in a number of points (where $\psi(q) = 1$ above) subject to the constraint that the value function in two neighboring states must not differ more than the cost of switching between those states.

The dual problem to find an upper bound was interpreted as minimizing the entire mass flow in the graph subject to the constraint that the mass production in each state of the system ($\psi(q)$ above) must not exceed the net flow out from that state.

A similar primal/dual problem can also be set up for a continuous system based on this reasoning. The discrete and continuous problems can then be combined to the hybrid version presented below (including the continuous problem as a special case).

Define a hybrid system as

$$\begin{cases} \dot{x}(t) = \mathbf{f}_{q(t)}(x(t), u(t)) \\ q(t) = \nu(x(t), q(t^-), \mu(t)) \end{cases} \tag{8}$$

where $x(t) \in X \subset \mathbf{R}^n$ is the state vector, $u(t) \in \Omega_u = Co\{u^1, u^2, \ldots, u^K\} \subset \mathbf{R}^m$ is a continuous input signal of the system. There is also a discrete

input, $\mu(t) \in \Omega_\mu$, which affects the evolution of the discrete variable $q(t) \in Q = \{1, 2, \ldots, N\}$. The notation $q(t^-)$ is used for the left-hand limit of q at t. $S_{q,r}$ is a set (parameterized by q and r) such that switching from mode q to r is possible when $x \in S_{q,r} \subseteq X$. The continuous state, x, is constrained to a hyperrectangle $X = \{x | \underline{c}_i \leq x_i \leq \overline{c}_i,\ \underline{c}_i \in \mathbf{R},\ \overline{c}_i \in \mathbf{R},\ i = 1, \ldots, n\}$.

The optimal control problem is to minimize the cost function

$$\int_{t_0}^{t_f} \mathbf{l}_q(x,u)dt + \sum_{k=1}^{k_f} s(x(t_k), q(t_k^-), q(t_k^+)) \tag{9}$$

subject to (8) while bringing the system from an initial state (x_0, q_0) at time t_0, to a final state (x_f, q_f) such that $x_f \in \Gamma_{q_f}$ at time t_f, where the end time, t_f, is free. Here, k_f is an arbitrary finite number of switches occurring at times $t_0 < t_1 < t_2 < \ldots < t_{k_f} < t_f$ and $s(x,q,r) > 0$ is an associated cost for switching from discrete state q to r, the continuous part being x just before the switch. Note that $s(\cdot) > 0$ excludes the possibility of infinitely many switches in an optimal solution. The final set is represented such that $\Gamma_q \subset X$ contains those x that are final in mode q. (If finishing in mode q is not allowed, then $\Gamma_q = \emptyset$.)

Sufficient conditions for a lower bound on the value function were given in [Hedlund and Rantzer, 1999]:

Let $V_q : X \to R$, $q \in Q$ be a set of continuous, piecewise $\mathcal{C}^1$ functions that satisfy

$$0 \leq \nabla V_q(x) \cdot \mathbf{f}_q(x,u) + \mathbf{l}_q(x,u) \qquad \forall x \in X,\ u \in \Omega_u,\ q \in Q \tag{10}$$

$$0 \leq V_r(x) - V_q(x) + s(x,q,r) \qquad \forall x \in S_{q,r}\ \ q,r \in Q : q \neq r \tag{11}$$

$$0 = V_q(x) \qquad \forall (x,q) | x \in \Gamma_q \tag{12}$$

where $\mathbf{f}_q(x,u)$ gives the dynamics of a hybrid system according to (8), $\mathbf{l}_q(x,u)$ and $s(x,q,r)$ define a cost function for the system according to (9). Then, for every (x_0, q_0), $V_{q_0}(x_0)$ gives a lower bound on the cost for optimally bringing the system from (x_0, q_0) to (x_f, q_f) such that $x_f \in \Gamma_{q_f}$.

3.1 Upper Bound on the Hybrid Value Function

One way of discretizing (10)–(12) to numerically obtain a lower bound of V_q was shown in [Hedlund and Rantzer, 1999]. The original inequalities were stated to give a lower bound on the optimal value function and the discretization was chosen to preserve this property, i.e. the solution to the discretized problem is in turn a lower bound on a function V_q that satisfies (10)–(12).

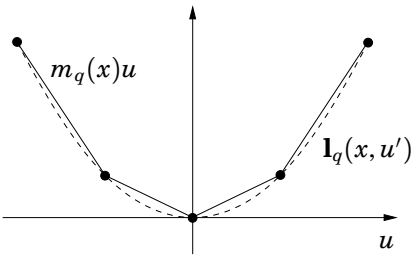


Figure 2. The solid line shows one possible approximation of u^2 (the dashed line) when $K = 5$.

It is desirable to estimate the approximation error for the discretized problem and to grasp the importance of various discretization parameters, e.g. the grid size. The dual problem formulation, that renders an upper bound on V_q, can give such insight.

To state the dual hybrid formulation, the following assumptions are made:

$$\mathbf{f}_q(x,u) = f_q(x) + g_q(x)u \tag{13}$$
$$\mathbf{l}_q(x,u) = l_q(x) + m_q(x)u \tag{14}$$

Note that these assumptions are not as restrictive as they might look at first glance. They allow any functions $\mathbf{f}_q(x,u)$ and $\mathbf{l}_q(x,u)$ to be approximated arbitrarily well.

EXAMPLE 4—APPROXIMATION OF A QUADRATIC COST FUNCTION
Consider the cost function $\mathbf{l}_q(x,u') = u'^2$ where $u' \in [-1,1]$. This function can be approximated by $m_q(x)u$, $u \in Co\{e_1, e_2, \ldots, e_K\}$, where $e_i \in \mathbf{R}^K$ is a unit vector in the direction of the i:th coordinate axis, $m_q(x) = [u_1^2, u_2^2, \ldots, u_K^2]$ is a row vector where $u_i = (2i - K - 1)/(K - 1)$.

The accuracy of the approximation increases with K. An example of $K = 5$ is shown in Fig. 2. □

THEOREM 1—UPPER BOUND ON THE INTEGRAL OF THE VALUE FUNCTION
Assume that $\rho_q^k : X \to \mathbf{R}^+$, $q \in Q$, $k = 1, 2, \ldots, K$ is piecewise $\mathcal{C}^1$ and $\lambda_{q,r} : X \to \mathbf{R}^+$, $(q,r) \in Q \times Q$, $q \neq r$ such that

$$0 = \rho_q^k(x) \qquad x \in \partial X,\ q \in Q,\ k = 1, 2, \ldots, K$$

and

$$\psi_q(x) \leq \sum_{k=1}^{K} \nabla \cdot \left(\rho_q^k(x)(f_q(x) + g_q(x)u^k)\right) + \sum_{r|x\in S_{q,r}} \lambda_{q,r}(x) - \sum_{r|x\in S_{r,q}} \lambda_{r,q}(x), \tag{15}$$

for all $(x, q) \in (X \backslash \Gamma_q) \times \mathcal{Q}$.

Then the following inequality holds for every $V_q : X \to R$, $q \in \mathcal{Q}$ satisfying (10)–(12) with $\mathbf{f}_q$ and $\mathbf{l}_q$ given by (13) and (14).

$$\begin{aligned} &\sum_q \int_{X\backslash\Gamma_q} \psi_q(x) V_q(x) dx \\ \leq &\sum_q \int_{X\backslash\Gamma_q} \left(\sum_{k=1}^{K} \rho_q^k(x)(l_q(x) + m_q(x)u^k) + \right. \\ &\left. + \sum_{r|x\in S_{q,r}} \lambda_{q,r}(x) s(x,q,r) \right) dx \end{aligned} \tag{16}$$

□

Remark 7. This theorem can be interpreted the same way as was done for the purely discrete case. Noting that the continuous control signal can be written as $u(x,q) = \sum_k \rho_q^k(x)u^k / \sum_k \rho_q^k(x)$, the inequality (15) corresponds to the mass production in state (x,q), $\psi_q(x)$ not exceeding the outflow (represented by the flow to other continuous states within the same discrete mode, ρ_q^k, and the flow to other modes, $\lambda_{q,r}$).

The inequality (16) shows that a summation of the value function is bounded from above by the cost of the overall flow in the dual setting.

Proof. Let $f_q^k(x) \equiv f_q(x) + g_q(x)u^k$ and $l_q^k(x) \equiv l_q(x) + m_q(x)u^k$. Then

$$\begin{aligned} &-\sum_{q\in\mathcal{Q}} \int_{X\backslash\Gamma_q} \left(\sum_{r|x\in S_{q,r}} \lambda_{q,r}(x) s(x,q,r) \right. \\ &\left. + \sum_{k=1}^{K} \rho_q^k(x) l_q^k(x) \right) dx \\ &+ \sum_{q\in\mathcal{Q}} \int_{X\backslash\Gamma_q} V_q(x)\psi_q(x) dx \end{aligned}$$

$$
\begin{aligned}
&\leq \sum_{q\in Q}\int_{X\setminus\Gamma_q}\left(\sum_{r|x\in S_{q,r}}\lambda_{q,r}\left(V_r(x)-V_q(x)\right)\right)dx\\
&\quad+\sum_{q\in Q}\int_{X\setminus\Gamma_q}\nabla V_q(x)\cdot\left(\sum_{k=1}^{K}\rho_q^k(x)f_q^k(x)\right)dx\\
&\quad+\sum_{q\in Q}\int_{X\setminus\Gamma_q}V_q(x)\left(\sum_{k=1}^{K}\nabla\cdot(\rho_q^k(x)f_q^k(x))\right.\\
&\quad\left.+\sum_{r|x\in S_{q,r}}\lambda_{q,r}(x)-\sum_{r|x\in S_{r,q}}\lambda_{r,q}(x)\right)dx\\
&=\sum_{q\in Q}\int_{X\setminus\Gamma_q}\nabla\cdot\left(V_q(x)\sum_{k=1}^{K}\rho_q^k(x)f_q^k(x)\right)dx\\
&\quad+\sum_{q\in Q}\sum_{r\in Q}\int_{(\Gamma_r\setminus\Gamma_q)\cap S_{q,r}}\lambda_{q,r}(x)V_r(x)dx\\
&\quad-\sum_{q\in Q}\sum_{r\in Q}\int_{(\Gamma_q\setminus\Gamma_r)\cap S_{q,r}}\lambda_{q,r}(x)V_r(x)dx\\
&=\sum_{q\in Q}\int_{\partial(X\setminus\Gamma_q)}\left(V_q(x)\sum_{k=1}^{K}\rho_q^k(x)f_q^k(x)\right)\cdot\mathbf{n}dS=0
\end{aligned}
$$

where the inequality above makes use of (10), (11), and (15).

Gauss' theorem is applied to the first equality on the last row ($\mathbf{n}$ is a unit vector that is orthogonal to $\partial(X\setminus\Gamma_q)$, pointing outwards from $X\setminus\Gamma_q$). Note that

$$\int_{(\Gamma_r\setminus\Gamma_q)\cap S_{q,r}}\lambda_{q,r}(x)V_r(x)dx=0,$$

since $V_r(x)=0$, $x\in\Gamma_r$ and that

$$\int_{(\Gamma_q\setminus\Gamma_r)\cap S_{q,r}}\lambda_{q,r}(x)V_r(x)dx=0,$$

since $\Gamma_q\cap S_{q,r}=\varnothing$ (switching is not allowed from a final state). □

4. Discretization

Utilizing a computer to solve (15) for a specific control problem, a straight forward approach is to grid the state space to require the inequality to

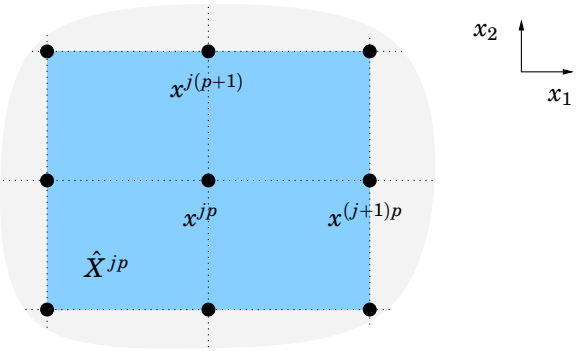


Figure 3. Illustration of X^{jp} and $\hat{X}^{jp}$.

be met at a set of uniformly distributed points in X. This approximation will, however, not guarantee an upper bound on the integral of the value function, unless the nature of $f_q(x)$ *between* the grid points is taken into consideration. This can be dealt with using a method similar to the one in [Hedlund and Rantzer, 1999].

For readability, discretization of a purely continuous system in a two-dimensional state space is presented below ($Q = \{1\}$, $n = 2$), i.e. the discrete mode subscript q and the mode switching terms containing λ vanish. Knowing how to handle the discretization of the continuous states, the discretization is easily extended to the hybrid case. Introduce the notation

$$\begin{aligned}
x^{jp} &= x_f + jhe_1 + phe_2 \\
X^{jp} &= \{x^{jp} + \theta_1 he_1 + \theta_2 he_2 : 0 \le \theta_i \le 1\} \\
\hat{X}^{jp} &= \{x^{jp} + \theta_1 he_1 + \theta_2 he_2 : -1 \le \theta_i \le 1\} \\
\rho^k(j,p) &= \rho^k(x^{jp}) \\
\Delta_i \rho^k(j,p) &= (\rho^k(x^{jp} + he_i) - \rho^k(x^{jp}))/h \\
\Delta_{-i} \rho^k(j,p) &= (\rho^k(x^{jp}) - \rho^k(x^{jp} - he_i))/h
\end{aligned}$$

where e_1 and e_2 are unit vectors along the coordinate axes, and h is the grid size. Define the $\min_{jp}$ operator such that

$$\min_{jp} f = \min_{x \in \hat{X}^{jp}} f(x)$$

and the $\max_{jp}$ operator analogously.

Also introduce new variables, $\alpha_i^k(j,p) \in \mathbf{R}$ and $\beta^k(j,p) \in \mathbf{R}$ for $k = 1,2,\ldots,K$, $i = 1,2$, and (j,p) such that $x^{jp} \in X \backslash \Gamma$. The inequality (15) can then be replaced by the following combination of backward and forward difference approximations that should hold for all $k = 1,2,\ldots,K$, $i = -2,-1,1,2$, and (j,p) such that $x^{jp} \in X \backslash \Gamma$:

$$\max_{jp}\psi \leq \sum_{k=1}^{K}\left(\sum_{i=1}^{2}\alpha_i^k(j,p) + \beta^k(j,p)\right) \tag{17}$$

$$\beta^k(j,p) \leq \min_{jp}\{\nabla\cdot(f+gu^k)\}\rho^k(j,p) \tag{18}$$

$$\beta^k(j,p) \leq \max_{jp}\{\nabla\cdot(f+gu^k)\}\rho^k(j,p) \tag{19}$$

$$\alpha_i^k(j,p) \leq \min_{jp}\{f_{|i|} + g_{|i|}u^k\}\Delta_i\rho^k(j,p) \tag{20}$$

$$\alpha_i^k(j,p) \leq \max_{jp}\{f_{|i|} + g_{|i|}u^k\}\Delta_i\rho^k(j,p) \tag{21}$$

For $x = x^{jp} + \theta_1 h e_1 + \theta_2 h e_2 \in X^{jp}$, $k = 1,2,\ldots,K$, define the interpolating functions

$$\begin{aligned}\rho^k(x) =& (1-\theta_1)(1-\theta_2)\rho^k(j,p)\\ &+ \theta_1(1-\theta_2)\rho^k(j+1,p)\\ &+ (1-\theta_1)\theta_2\rho^k(j,p+1)\\ &+ \theta_1\theta_2\rho^k(j+1,p+1)\end{aligned} \tag{22}$$

The following result applies.

THEOREM 2—DISCRETIZATION IN $\mathbf{R}^2$
If $Q = \{1\}$ and $\rho^k(j,p)$ satisfy (17)–(21) for all grid points $x^{jp} \in X \subset \mathbf{R}^2$ such that X^{jp} intersects X, then the interpolating functions $\rho^k(x)$ defined by (22) satisfies (15) and an upper bound of $\int_{X\backslash\Gamma} V(x)\psi(x)dx$ is given by (16). □

Applying the above discretization scheme to a couple of examples, the resulting problem often seem to be ill conditioned. The reason for this is likely the inequalities (17)–(21) being to conservative. Other parameterizations of ρ_q^k may give better results.

5. Summary

We have derived an inequality which is dual to the "Hybrid Bellman inequality" presented in an earlier paper. The dual optimization problem has a simple physical interpretation in terms of particle flows. For a given

control law one should envision particles flowing along the system trajectories everywhere in the state space. In a steady state situation, with particle production everywhere, the concentration of particles must be infinite near the equilibrium. The dual linear programming problem is stated in terms of the particle concentrations. It can be viewed as a generalization of the classical flow problems in discrete optimization to the case of hybrid systems.

The dual gives an upper bound on the optimal cost and thus contains valuable information about the conservatism introduced in the discretization of the primal problem. A discretization scheme that preserves the upper bound property has been proposed. Numerical problems call for further research on alternate discretization.

Paper E

A Toolbox for Computational Analysis of Piecewise Linear Systems

Sven Hedlund and Mikael Johansson

Abstract

This paper reports the development of a Matlab toolbox for computational analysis of piecewise linear systems. The analysis is based on piecewise quadratic Lyapunov functions, which are computed via convex optimization. In this way, exponential stability and system performance can be assessed. The toolbox also supports efficient simulation of systems with discontinuous dynamics and sliding modes. A set of intuitive commands for describing piecewise linear systems is included, making the analysis routines easily accessible also for the inexperienced user.

Keywords piecewise linear system, LMIs, toolbox

Reprinted from: Hedlund, S. and M. Johansson (1999): "A Toolbox for Computational Analysis of Piecewise Linear Systems", *Proceedings of European Control Conference*, with permission from EUCA.

1. Introduction

As performance demands on modern control systems increase, controllers are required to work over large operating ranges where assumptions on linear dynamics are no longer valid. Successful design and tuning of such controllers are strongly dependent on the possibility of analyzing the effects that arise away from equilibrium conditions. An interesting class for studying such problems is the class of piecewise linear systems. It captures the effects of saturations and state constraints, and is also a good candidate for studying hybrid control systems (cf. [Sontag, 1996]). Moreover, many popular control schemes, such as gain scheduling and fuzzy logic controllers, can be well modeled by piecewise linear systems (cf. [Årzén *et al.*, 1998]).

Recently, it has been shown how stability and performance of piecewise linear systems can be assessed using Lyapunov functions that are piecewise quadratic [Joahnsson and Rantzer, 1996]. Such Lyapunov functions can be computed via convex optimization in terms of linear matrix inequalities (LMIs). The approach gives a drastic reduction of conservatism compared to approaches based on a single quadratic Lyapunov function[Corless, 1994], while computations remain comparatively efficient.

This paper gives an overview of a MATLAB toolbox for computational analysis of piecewise linear systems. The main purpose of the paper is to show how simple the toolbox, PWL Tool, makes experimenting with piecewise linear systems. For a detailed description of usage, the reader is referred to the manual [Hedlund and Johansson, 1999].

PWL Tool is available free of charge upon request from the authors.

2. Model Representation

The toolbox handles piecewise affine systems on the form

$$\begin{cases} \dot{x} = A_i x + a_i + B_i u \\ y = C_i x + c_i + D_i u \end{cases} \qquad \text{for } x \in X_i. \tag{1}$$

Here, $\{X_i\}_{i \in I} \subseteq \mathbf{R}^n$ is a partition of the state space into a number of closed (possibly unbounded) polyhedral cells (cf. e.g. Fig. 1), and I is the index set of the cells. In order to allow rigorous analysis of smooth nonlinear systems, the toolbox allows the system dynamics to lie in the convex hull of a set of piecewise affine systems, see [Johansson, 1999]. This is e.g. useful for the analysis of fuzzy Takagi-Sugeno systems.

For convenient notation, we introduce

Table 1. Commands for building a PWL system.

command	description
setpwl	initialize PWL object
addregion	define polyhedral region
addynamics	define system dynamics
getpwl	extract PWL object

$$\bar{A}_i = \begin{bmatrix} A_i & a_i \\ 0 & 0 \end{bmatrix} \qquad \bar{C}_i = [\, C_i \quad c_i \,] \qquad \bar{x} = \begin{bmatrix} x \\ 1 \end{bmatrix}$$

A large part of the analysis results will be concerned with (global) properties of equilibria. We therefore let $I_0 \subseteq I$ be the set of indices for the cells that contain the origin, and $I_1 \subseteq I$ be the set of indices for cells that do not contain the origin. We will assume that $a_i = 0$, $c_i = 0$ for $i \in I_0$.

The cells are represented by matrices $\bar{G}_i$ that satisfy

$$\bar{G}_i \bar{x} \succeq 0, \qquad \text{if and only if } x \in X_i \tag{2}$$

Here, the vector inequality $z \succeq 0$ means that each entry of z is nonnegative. We recognize this as the halfspace representation of a polyhedron. It is also necessary to specify matrices $\bar{F}_i = [\, F_i \quad f_i \,]$ with $f_i = 0$ for $i \in I_0$ that satisfy

$$\bar{F}_i \bar{x} = \bar{F}_j \bar{x} \qquad \text{for } x \in X_i \cap X_j. \tag{3}$$

These matrices are used to parameterize the Lyapunov function candidate to be continuous across cell boundaries. The PWL Tool handles a piecewise linear (PWL) system as an object. The basic commands for building a PWL system are listed in Table 1. Having partitioned the state space and used the functions for entering data into MATLAB, the system is aggregated into a single record that is passed on to functions for analysis and simulations.

The command `setpwl` initializes the PWL object and should be run first. When this is done, one will typically define the entire system by repeatedly calling `addynamics` and `addregion`. The command `addynamics` is used to specify the matrix variables (as given by (1)) corresponding to the dynamics in a certain region of a PWL system. An identifier is returned for future reference to the dynamics. The command `addregion` lets the user enter the region specific data ($\bar{G}_i$- and $\bar{F}_i$-matrices) and via the references returned by `addynamics` specify the dynamics in the region. By specifying several system matrices in one region, one indicates that

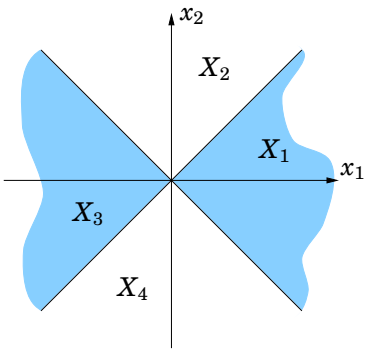


Figure 1. Partitions of the flower example.

the dynamics lies in the convex hull of these systems. When all matrices are entered, the PWL object is extracted by `getpwl`. In addition to linking several dynamics to one region, it is also possible to link several regions to the same dynamics. (This could sometimes be useful to save some data space and typing effort.)

EXAMPLE 1—THE FLOWER SYSTEM
The following system, whose partition is illustrated in Figure 1, has been used in [Joahnsson and Rantzer, 1996] in order to demonstrate the flexibility of piecewise quadratic Lyapunov functions.

$$\dot{x} = \begin{cases} A_1 x = \begin{bmatrix} -0.1 & 1 \\ -5 & -0.1 \end{bmatrix} x & x \in X_1 \cup X_3 \\ A_2 x = \begin{bmatrix} -0.1 & 5 \\ -1 & -0.1 \end{bmatrix} x & x \in X_2 \cup X_4 \end{cases}$$

The following lines of code defines the “flower system”.

```
% Initialize the PWL object
  setpwl([]);
% Enter A-matrices
  A1 = [-0.1 1; -5 -0.1];
  A2 = [-0.1 5; -1 -0.1];
% Set up dynamics
  d1 = addynamics(A1);
  d2 = addynamics(A2);
% Enter G- and F-matrices
  G1 = [ 1  1 0; -1  1 0];
```

```
  G2 = [ 1 -1 0;  1  1 0];
  G3 = [-1 -1 0;  1 -1 0];
  G4 = [-1  1 0; -1 -1 0];
  F1 = ...
  F2 = ...
  ...
% Define cells
  addregion(G1, F1, d1);
  addregion(G2, F2, d2);
  addregion(G3, F3, d1);
  addregion(G4, F4, d2);
% Extract PWL object
  pwlsys = getpwl;
```

We will return to this example later to assess global exponential stability of the origin. □

3. Describing Polyhedral Partitions

Defining all the data that the computational engine of PWL Tool needs can be far from easy for the inexperienced user. It is therefore desirable to relieve the user from this task. In this section, we describe a set of user-friendly commands for specifying piecewise linear systems that automatically computes the constraint matrices, G_i and F_i, used by PWL Tool.

The toolbox currently supports partitions induced by global hyperplanes and simplex partitions (see [Johansson, 1999] for precise definitions, and more elaborate explanations), but the layered structure of the toolbox makes it easy to add support for other types of partitions.

3.1 Describing Hyperplane Partitions

Specifying a hyperplane partition essentially consists of defining the generating hyperplanes, introducing the cells by stating which generating hyperplanes that bound the cell, and giving the affine dynamics valid within each region. Table 2 specifies a number of commands that support these steps.

The command `setpart` initializes a new partition, and should be issued prior to defining the partition components. In order to indicate the type of partition, `setpart` takes the argument `'h'` for hyperplane partitions and `'s'` for simplex partitions.

The commands `addhp` and `addati` define generating hyperplanes and affine dynamics respectively. Both commands return an identifier for later

Table 2. Commands for defining hyperplane partitions.

Command	Description
setpart	Initialize partition data structure
addhp	Add hyperplane
addati	Specify affine dynamics
addhcell	Define hyperplane cell
getpart	Retrieve partition data structure
part2pwl	Convert to generic data structure

reference. Cells are subsequently defined using the command `addhcell`, which takes two arguments. The first argument specifies the bounding hyperplanes (using their identifiers returned by `addhp`), and the second argument specifies the dynamics valid in the region (using the identifiers returned by `addati`). The sign of the hyperplane reference indicates on "what side" of the hyperplane the cell is located.

The command `getpart` returns a data structure that describes the partition. Finally, the command `part2pwl` computes the data required by the computational engine of PWL Tool. The computations performed by `part2pwl` are explained in [Joh99]

We illustrate the commands on a simple relay feedback system.

EXAMPLE 2—A RELAY FEEDBACK SYSTEM
Consider a linear system under relay feedback

$$\begin{cases} \dot{x} &= Ax + Bu \\ y &= Cx \\ u &= -\text{sign}(y). \end{cases}$$

The relay feedback induces a piecewise linear system with two regions, separated by the switching hyperplane $Cx = 0$. The following lines of MATLAB code define the relay system using hyperplane partitions.

```
% Initialize hyerplane partition
  setpart('h');
% Define boundary hyperplanes
  switch_plane = addhp([C 0]);
% Dynamics \dot{x}=Ax+B and \dot{x}=Ax-B
  d_on = addati(A,-B);
```

Table 3. Commands for defining simplex partitions.

Command	Description
setpart	Initialize partition data structure
addvtx	Add vertex
addray	Add ray
addati	Specify affine dynamics
addscell	Define simplex cell
getpart	Retrieve partition data structure
part2pwl	Convert to generic data structure

```
  d_off= addati(A,B);
% Introduce cells
  X_1 = addhcell(switch_plane, d_on);
  X_2 = addhcell(-switch_plane, d_off);
% Retrieve data structure
  part = getpart;
% Transform to PWL data structure
  pwlsys = part2pwl(part);
```

□

3.2 Describing Simplex Partitions

The specification of a simplex partition is very similar to the definition of a hyperplane partition. The main difference is that (generalized) simplices are defined by vertices ("points") and rays ("directions") rather than the equations for its bounding hyperplanes (cf. [Johansson, 1999]). The commands for building simplex partitions are shown in Table 3. A new simplex partition is initialized by the command `setpart('s')`. A (generalized) simplex in R^n is defined by $n+1$ vertices or rays. Vertices and rays are defined by the commands `addvtx` and `addray`. Both commands return an identifier for later reference. As in the hyperplane case, the dynamics are defined by the command `addati`. The cells of the partition are defined by the command `addscell`, which takes three arguments. The first two arguments are lists of vertex and ray references respectively, while the last argument specifies the dynamics valid within the region. The total number of vertex and ray references sums to $n+1$, and at least one extreme point of the cell is a vertex. Once all cells are defined, the command `getpart` retrieves a data structure describing the partition, and

the command `part2pwl` transform this information into the data required by `pwltools`. We return to Ex. 1 to demonstrate the commands.

EXAMPLE 3—FLOWER SYSTEM – SIMPLEX DESCRIPTION

```
% Initialize simplex partition
  setpart('s');
% Define vertices and rays
  v1 = addvtx([0 0]);
  r1 = addray([1 1]);
  r2 = addray([-1 1]);
  r3 = addray([-1 -1]);
  r4 = addray([1 -1]);
% Set-up dynamics
  d1 = addati(A1);
  d2 = addati(A2);
% Define cells
  X_1 = addvcell([v1],[r1 r2],d1);
  X_2 = addvcell([v1],[r2 r3],d2);
  X_3 = addvcell([v1],[r3 r4],d1);
  X_4 = addvcell([v1],[r4 r1],d2);
% Retrieve partition data structure
  part = getpart;
% Transform into PWL Tool data structure
  pwlsys = part2pwl(part);
```

□

4. Simulation of Piecewise Linear Systems

Simulation is one of the most important tools for evaluating new control strategies, in academia as well as in industry. Although there has been a strong development of general-purpose simulation environments during the last 20 years, simulation of systems with switching and discontinuous dynamics is still poorly supported by most software packages. In the context of piecewise linear systems, problems may occur when the vector fields are discontinuous across cell boundaries. If the flow in two neighboring cells point toward their common boundary, cf. Fig. 2, the state goes through a number of infinitely fast mode changes that cause most simulators to 'get stuck'. The nature of these fast mode changes has been studied by several researchers, see [Filippov, 1988; Utkin, 1977]. In general, the net effect of the fast mode switches is a constrained motion

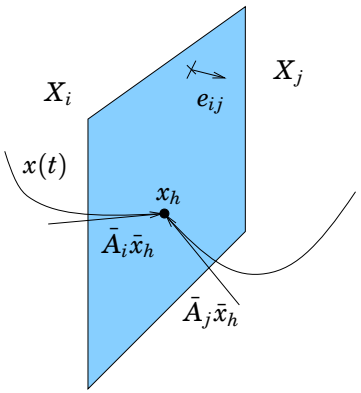


Figure 2. Sliding mode surface.

along the switching surface, referred to as a sliding motion. The dynamics of the sliding motion can be uniquely defined for simple boundaries, while intersecting boundaries may cause uniqueness problems. Figure 3 gives an overview of how PWL Tool handles simulations. Before starting, some preparatory computations are made. During the initialization phase, [1], each region is assigned a number of pointers to the neighboring regions to allow for efficient switching. In addition, each surface separating the regions undergo sliding mode analysis. Define e_{ij} to be the normal vector of the hyperplane between X_i and X_j directed from X_i to X_j, cf. Fig. 2. The surface then contains a sliding mode if there exist an x such that

$$\begin{aligned} \bar{G}_i\bar{x} &\succeq 0 \\ \bar{G}_j\bar{x} &\succeq 0 \\ e_{ij}^T\bar{A}_i\bar{x} &> 0 \\ -e_{ij}^T\bar{A}_j\bar{x} &> 0 \end{aligned} \tag{4}$$

This is an LP problem, the result of which is patched up for each boundary into one single matrix. The first step of the actual simulation is to find the initial region, [2], i.e. if starting in x_0, find i such that $\bar{G}_i\bar{x}_0 \succeq 0$. During the first visit to [2], the G_i-matrices have to be tested one by one. Thanks to the initialization phase, however, this is avoided when entering next time. Having found the right region, the simulation is started, [3], and proceeded until the boundary is hit. When a boundary is hit, one must check whether to enter the sliding mode state, [4]. This is done by first looking up into the sliding mode matrix whether the surface contains a sliding mode. If it does, the conditions (4) are checked for the specific entry

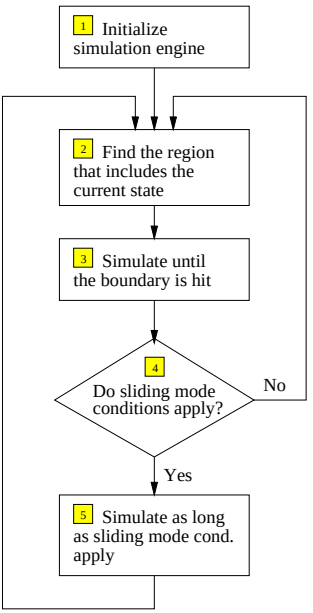


Figure 3. Schematic description of simulation algorithm for systems with sliding modes.

point. Having entered the sliding mode state, 5, The resulting equivalent dynamics is computed according to Filippov's convex definition [Filippov, 1988]:

$$\dot{x} = \begin{cases} \bar{A}_i \bar{x}, & x \in X_i \\ \bar{A}_j \bar{x}, & x \in X_j \\ \lambda(x)\bar{A}_i \bar{x} + (1-\lambda(x))\bar{A}_j \bar{x}, & x \in X_i \cap X_j \end{cases}$$

where $\lambda(x)$ is the solution to

$$e_{ij}^T \left(\lambda(x)\bar{A}_i \bar{x} + (1-\lambda(x))\bar{A}_j \bar{x}\right) = 0$$

Currently, PWL Tool does not support sliding mode on intersecting hyperplanes.

Table 4 lists the commands that are available for detecting sliding modes and simulating PWL systems with sliding modes. The command `findsm` searches all the boundaries between cells and informs the user of between which cells sliding modes are possible. This is of course interesting from a stability analysis point of view. Knowing that most uncontrolled

Table 4. Simulation related commands.

command	description
findsm	detect sliding modes
pwlsim	simulate PWL system

systems do not exhibit sliding modes, however, this command can sometimes give a first warning if the system is not modeled in an appropriate way.

A trajectory can be simulated from a given initial state with `pwlsim`. The outputs from this function are, in addition to the time vector and matching state vectors, the times when cell switching occurred and the corresponding cells that have been visited.

EXAMPLE 4—RELAY SYSTEM WITH SLIDING MODE
Returning to the relay feedback system of Ex. 2, we now consider the following non-minimum phase system from [Johansson, 1997]:

$$\dot{x} = \begin{bmatrix} -3 & 1 & 0 \\ -3 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x$$

It is assumed that `A` and `B` have been entered before executing the code of Ex. 2.

```
% Search for sliding modes
  findsm(pwlsys);
    Sliding mode detected on boundary between
    cell 1 and 2.
% Simulate the system
  x0 = [1 -1 0]';
  [t, x, te] = pwlsim(pwlsys, x0, [0 20]);
```

The above code establishes that the system exhibits a sliding mode on the switching surface. Simulating the system using the command `pwlsim`, one can see how the system tends to a limit cycle with sliding mode, see Figure 4. □

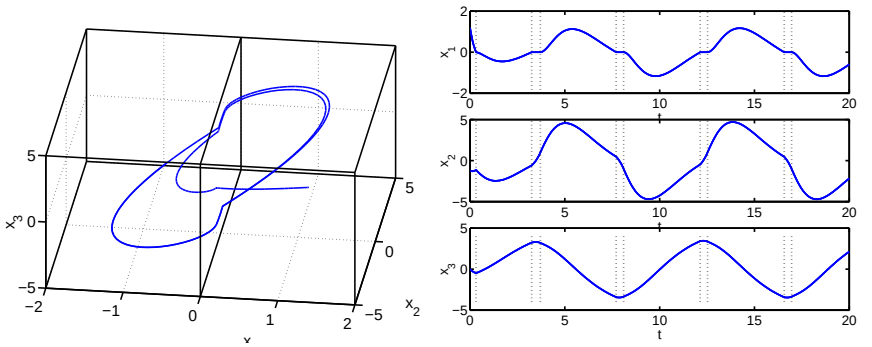


Figure 4. Limit cycle with sliding trajectory. The vertical dashed lines in the right part indicate time instances for the mode selection.

Table 5. Commands for stability analysis.

command	description
qstab	quadratic stability analysis
pqstab	piecewise quadratic analysis
pqstabs	d.o. taking sliding into account

5. Computation of Piecewise Quadratic Lyapunov Functions

In PWL Tool, stability of PWL systems is proved with the aid of piecewise quadratic (PWQ) Lyapunov functions. This is less conservative than the commonly used global quadratic approach and the toolbox makes it possible to prove stability for PWL systems that do not admit quadratic Lyapunov functions.

The F-matrices as defined by Eq. (3) are used to force continuity of the Lyapunov function. It is parameterized by a symmetric matrix, T, as follows

$$V(x) = \bar{x}^T \bar{F}_i^T T \bar{F}_i \bar{x} \qquad x \in X_i, \qquad i \in I.$$

This structure allows the usual constraints on $V(x)$ (positive definiteness and decrement along the system trajectories) to be expressed as a set of LMIs [Johansson, 1999].

The commands provided for stability analysis are shown in Table 5. The command `pqstab` searches for a PWQ Lyapunov function as described above. If there exist a piecewise quadratic Lyapunov function, `pqstab`

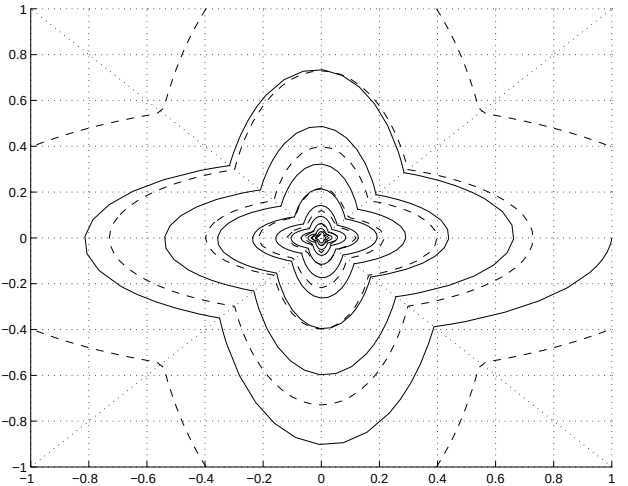


Figure 5. Simulation of the flower system (solid line) and level curves of a PWQ Lyapunov function (dashed).

returns a three dimensional array, a vector of matrices, where matrix no. i corresponds to $\bar{F}_i^T T \bar{F}_i$ of eq. (5). The command `qstab` tries to find a *global* quadratic Lyapunov function ($V(x) = x^T P x$). This is of course conservative, but `qstab` uses the state space partitioning structure to relax the constraints on the Lyapunov function. In addition, the simplicity of a globally quadratic function often makes it a natural choice for a first attempt.

The LMI:s stated in `pqstab` for the decreasing condition are only valid for systems without any sliding modes. The command `pqstabs` is slightly modified to be able to handle the sliding mode case.

EXAMPLE 5—FLOWER SYSTEM — STABILITY ANALYSIS
We now try to prove stability of the flower system (Ex. 1).

```
% Search for sliding modes
  findsm(pwlsys);
  There are no sliding modes.
% Since there are no sliding modes, use pqstab
  pqstab(pwlsys);
  Lyapunov function was found.
```

Level surfaces of the Lyapunov function are plotted together with a simulated trajectory (using `pwlsim` as shown in Ex 4) in Fig. 5. □

Table 6. Commands for performance analysis and control design.

command	description
iogain	L_2 gain computation
pqobserv	Output energy estimation
optcstlb	Lower cost for LQG problem
pwlctrl	Derive piecewise LQG controller
optcstub	Estimate cost achieved by pwlctrl

6. Performance Analysis and Control Design

Having utilized the Lyapunov function machinery for assessing stability, it can be used in a similar way for other computations. PWL Tool supports performance analysis and control design. Table 6 lists the commands available. All of these commands estimate an upper and/or a lower bound on a certain performance property. If the estimates are too coarse, the results can be refined by further refinement of the state space partitions. The command `iogain` computes an upper bound on the L_2 induced input output gain of a PWL system. The command `pqobserv` computes a lower and an upper bound on the integral of the output energy for a given initial state, $x(0)$.

There are three commands related to controller synthesis. The optimal control problem for piecewise linear systems (while bringing the system to $x(\infty) = 0$ from an arbitrary initial state, $x(0)$) can be defined to minimize the cost

$$J(x_0, u) = \int_0^\infty \left(\bar{x}^T \bar{Q}_{i(t)} \bar{x} + u^T R_{i(t)} u\right) dt$$

(Here $i(t)$ is defined so that $x(t) \in X_{i(t)}$.) A lower bound, on the minimum achievable J is computed by `optcstlb` The command `pwlctrl` creates a PWL controller based on the results from `optcstlb`. A vector of matrices representing the state feedback used in different regions is returned. Having applied the controller given from above, `optcstub` returns an upper bound on the resulting optimal cost.

7. Summary

This paper has presented a MATLAB toolbox for analysis of piecewise linear systems, a class of nonlinear systems that appears frequently in control

theory, e.g. in hybrid systems and linear systems with various constraints. The analysis is based on piecewise quadratic Lyapunov functions, which are computed via convex optimization. In this way, exponential stability and system performance can be assessed for this class of nonlinear systems. The toolbox also supports efficient simulation of systems with discontinuous dynamics and sliding modes.

PWL Tool makes it simple to experiment with piecewise linear systems. The authors provide it free of charge upon request, with a reference manual [Hedlund and Johansson, 1999] and additional examples.

Acknowledgments

This work was supported by the Esprit LTR project FAMIMO and TFR project 95-759.

2

A MATLAB Tool for Dynamic Programming for Hybrid Systems

2.1 Introduction

This chapter presents "CDP Tool", a set of MATLAB commands for optimal control via convex dynamic programming of a class of hybrid systems.

The presentation is organized as follows: Section 2.2 defines the main problem that CDP Tool is designed for. Section 2.3 gives an overview of the CDP Tool commands and the main ideas behind the computations, the purpose being to give insight into the possibilities and limitations of CDP Tool. For a more comprehensive presentation of the underlying theory, see Paper A of this thesis. For a complete description of the commands, see the reference manual in [Hedlund, 1999]. Section 2.4 demonstrates the usage on some examples.

2.2 Problem Formulation

Let $X \subset \mathbf{R}^n$ and $\Omega_u \subset \mathbf{R}^p$ denote a continuous state space and input space respectively. Let the finite set $\bar{Q}$ denote a discrete state space as well as an input space. Time driven dynamics is governed by the function $f : X \times \bar{Q} \times \Omega_u \to \mathbf{R}^n$

The hybrid system model is defined via the trajectories it accepts. Given the function $u : [t_0, t_M] \to \Omega_u$, an increasing sequence of $M + 1$ real numbers, $T = (0 = t_0, t_1, \ldots, t_M \leq \infty)$, and the piecewise constant

function $\mu : [t_0, t_M] \to \bar{Q}$ such that $\mu(t) = \mu_k \in \bar{Q}$ for $t \in [t_k, t_{k+1})$, a trajectory of the hybrid system

$$\begin{aligned} \dot{x}(t) &= f(x(t), q(t), u(t)) && (2.1) \\ q(t) &= \mu(t) && (2.2) \end{aligned}$$

is defined as the collection (T, x, q) where $x : [t_0, t_M] \to X$ is absolutely continuous and satisfies (2.1) for $t \in (t_k, t_{k+1})$, $k = 0, 1, \ldots, M-1$ and $q : [t_0, t_M] \to \bar{Q}$ satisfies (2.2). The time argument, t, will often be omitted in the sequel for readability.

Given the nonnegative functions $l : X \times \bar{Q} \times \Omega_u \to \mathbf{R}$ and $s : X \times \bar{Q} \times \bar{Q} \to \mathbf{R}$ with $s(\cdot, \cdot, \cdot) \geq \varepsilon > 0$, the initial state $(x_0, q_0) \in X \times \bar{Q}$, and the final set $\Gamma \subset X \times \bar{Q}$, the optimal control problem is to minimize the cost functional

$$J(x_0, q_0, u(\cdot), \mu(\cdot)) = \int_0^{t_M} l(x, q, u) dt + \sum_{k=1}^{M} s(x(t_k), q(t_k^-), q(t_k^+)) \qquad (2.3)$$

over $u(\cdot)$, M, and $\mu(\cdot)$, subject to (2.1), (2.2), $(x, q)(0) = (x_0, q_0)$, and $(x, q)(t_M) \in \Gamma$.

Hence, l is a function penalizing continuous evolution and s penalizes switches. The requirement that $s \geq \varepsilon > 0$ makes CDP Tool exclude solutions with an infinite number of switches. Sec. 2.3 will show that the tool also handles exponential time weighting of the cost function.

It can be noted that the dynamics of (2.2) is less general than the event driven dynamics of the hybrid system model (1.1) in Chapter 1, where $q(t) = \nu(x(t), q(t^-), \mu(t))$. The role of the ν function, however, preventing certain mode switches from certain parts of the state space while enforcing switches from other parts, can be achieved in CDP Tool by appropriate construction of the penalty functions l and s.

2.3 Features of CDP Tool

The main commands available for solving the control problem are listed in Table 2.1. The purpose of these commands will be explained in this section by means of a brief presentation of the underlying theory. For a more comprehensive presentation of the theory, see Paper A of this thesis.

Most of the MATLAB commands that constitute CDP Tool can be divided into three groups; value function computation, control feedback law computation, and simulation. The basis for solving the optimal control

Table 2.1 The essential MATLAB commands of CDP Tool.

Command	Description
cdplows	Compute a lower bound of the value function, single-point maximization
cdplowes	Compute a lower bound of the value function of an exponential time weighting problem, single-point maximization
cdplowm	Compute a lower bound of the value function, multi-point maximization
cdplowem	Compute a lower bound of the value function of an exponential time weighting problem, multi-point maximization
cdpctrl	Compute a control signal, based on an approximation of the value function
cdpsim	Simulate controlled system
cdpsimf	Simulate controlled system, fixed time step
cdpsime	Simulate controlled system with exponential time cost function
cdpsimef	Simulate controlled system with exponential time cost function, fixed time step

problem in CDP Tool is the optimal value function, $V^\star : X \times \bar{Q} \to \mathbf{R}$, defined as

$$V^\star(x_0, q_0) = \min_{u(\cdot),\mu(\cdot)} J(x_0, q_0, u, \mu) \tag{2.4}$$

CDP Tool makes use of the property derived in Paper A, that any function $V : X \times \bar{Q} \to \mathbf{R}$ that satisfies

$$0 \leq \frac{\partial V(x,q)}{\partial x} f(x,q,u) + l(x,q,u) \qquad \forall (x,q) \in (X \times \bar{Q}) \setminus \Gamma,\ u \in \Omega_u \tag{2.5}$$

$$0 \leq V(x,r) - V(x,q) + s(x,q,r) \qquad \forall x \in X,\ q,r \in Q,\ q \neq r \tag{2.6}$$

$$0 = V(x,q) \qquad (x,q) \in \Gamma \tag{2.7}$$

is a lower bound of $V^\star(x,q)$. Since the inequalities (2.5)-(2.7) are linear constraints in $V(x,q)$, maximization of $V(x_0,q_0)$ subject to the inequalities

is a linear program (LP). Notes on how to discretize this problem to be able to solve it with a computer will be presented after the following observation.

Exponential Time Weighting

Value function constraints that are similar to the ones above can also be used for problems with exponential time weighting of the cost function. Define the cost function $J_e(x_0, q_0, u(\cdot), \mu(\cdot))$ as

$$\begin{aligned} J_e(x_0, q_0, u(\cdot), \mu(\cdot)) &= \int_0^\infty l_e(x, q, u, t)dt + \sum_{k=1}^{M} s_e(x(t_k), q(t_k^-), q(t_k^+), t_k) \\ &= \int_0^\infty \tilde{l}(x, q, u)e^{-at}dt + \sum_{k=1}^{M} \tilde{s}(x(t_k), q(t_k^-), q(t_k^+))e^{-at_k} \end{aligned} \quad (2.8)$$

where $a > 0$. The rest of the parameters is defined analogously to (2.3). If $V_e^\star(x, q, t)$ is defined as the optimal cost for starting in (x, q) at time t, then the time dependent lower bound counterpart of (2.5) can be written

$$\frac{\partial V_e(x, q, t)}{\partial t} + \frac{\partial V_e(x, q, t)}{\partial x} f(x, q, u) + l_e(x, q, u, t) \geq 0 \quad (2.9)$$

Rewriting the functions like $V_e(x, q, t) = e^{-at}\tilde{V}_e(x, q)$ and $l_e(x, q, u, t) = \tilde{l}_e(x, q, u)e^{-at}$, (2.9) becomes

$$-a\tilde{V}_e(x, q) + \frac{\partial \tilde{V}_e(x, q)}{\partial x} f(x, q, u) + \tilde{l}_e(x, q, u) \geq 0 \quad (2.10)$$

Thus, the time dependence introduced in the value function cancels and a slight variation of (2.5)—(2.7) will give the corresponding LP for the problem with exponential time weighting.

Discretization

Using a computer to find a value function that satisfies (2.5)-(2.7) for a specific control problem, a straightforward approach is to grid the state space to require the inequalities to be satisfied for a set of uniformly distributed points in X. Let $e_1, e_2, \ldots, e_n$ denote the unit vectors along the coordinate axes and define the discretization vector $h \in \mathbf{R}^n$ such that h_i (the i:th component of h) is the distance between the grid points in the direction of e_i. A small part of a discretization in $\mathbf{R}^2$ around a grid point x_p is shown in Fig. 2.1.

Each of the value function commands (`cdplows`, `cdplowes`, `cdplowm`, and `cdplowem`) applies a discretization grid like this to X. The commands

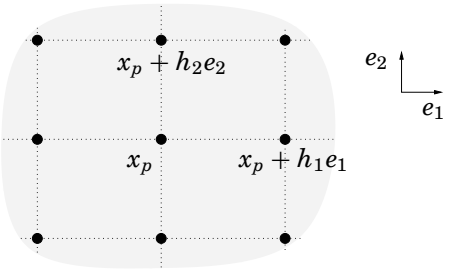


Figure 2.1 Illustration of the discretization grid in $\mathbf{R}^2$.

handle sets, X, that are hyperrectangles in $\mathbf{R}^n$, and the user specifies the granularity of the grid by the input vector $N \in \mathbf{Z}^n$ such that the k:th component of N is the number of grid points in the direction of e_k.

This way, the function V, to be computed to be a lower bound of the optimal value function, is parameterized by its values in the gridpoints and the values between the gridpoints are obtained by multilinear interpolation. For (2.5) to hold in the entire state space rather than in the gridpoints solely, the discretization must be constructed to also consider the values of f and l *between* the gridpoints. The value function commands of CDP Tool can be set up to use a method (presented in Paper A) for preserving the lower bound property. For each grid point, x_p, this method requires the extremal values of $f(x_p, q, u)$ and $l(x_p, q, u)$ in a neighborhood of x_p as follows.

Define the hyperrectangle $\hat{X}^p$ surrounding grid point x_p as

$$\hat{X}^p = \{x_p + \sum_{i=1}^{n} \theta_i h_i e_i : -1 \leq \theta_i \leq 1\}. \tag{2.11}$$

An illustration of this set in a two dimensional space is shown in Fig. 2.2. For each grid point, x_p, the value function commands then need

$$\underline{f}^p(q,u) = \min_{x \in \hat{X}^p} f(x,q,u) \tag{2.12}$$

$$\overline{f}^p(q,u) = \max_{x \in \hat{X}^p} f(x,q,u) \tag{2.13}$$

$$\underline{l}^p(q,u) = \min_{x \in \hat{X}^p} l(x,q,u) \tag{2.14}$$

to form the discretized inequalities. (The extrema should be computed component wise in the vectors.) Note, however, that the CDP Tool value function commands also can be called without requiring the true bound

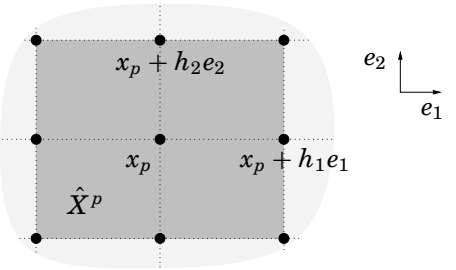


Figure 2.2 Illustration of $\hat{X}^p$ in $\mathbf{R}^2$.

property, often resulting in a plausible value function without the sometimes cumbersome search for local extrema.

The extremal values of (2.12)-(2.14) still depend on the continuous parameter u. The continuous control signal has to be discretized by the CDP Tool user in a tradeoff between accuracy and computational speed. The reason for leaving this burden to the user instead of automatically gridding the problem in u (as was done for x), is that insight in the structure of the problem might be used for clever gridding and a reduction of the computational load. Consider e.g. the analysis of a system with $\Omega_u = [-1, 1]$. A standard gridding may lead to the approximation $\Omega_u = \{-1, -0.8, -0.6, \ldots, 1\}$. If the system dynamics is affine in u and the optimization criterion is minimum time, however, the resulting control law is bang-bang and the obvious choice is $\Omega_u = \{-1, 1\}$.

Single-point vs. Multi-point Maximization

Instead of computing the value function in one single point, (x_0, q_0), it is often desirable to get an estimate for the value function in a larger subset of X. Some of the CDP Tool value function commands do this by maximizing the sum of $V(x, q)$ in several grid points, which is called multi-point maximization. The region that contains the grid points for which V is maximized, the "optimization region", is denoted $O \subset X$. Fig. 2.3 illustrates the two maximization alternatives in $\mathbf{R}^2$ with some sample trajectories.

There is in general no guarantee that a region maximization will give the same result as a single point maximization, i.e. maximizing $V(x_0, q_0)$ solely may give a different result from the value of $V(x_0, q_0)$ coming from a region maximization where $(x_0, q_0) \in O$. Experience from examples, however, tells that the difference is often small, making the benefit of receiving the value function in a large region solving *one* LP an attractive alternative.

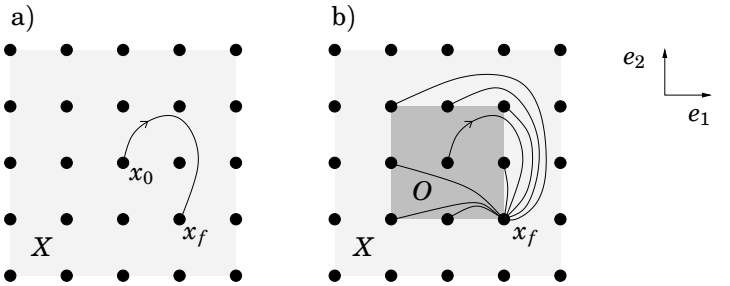


Figure 2.3 Various choices of maximization in $\mathbf{R}^2$ with corresponding optimal state trajectories. a) Single point maximization. b) Region maximization.

The sample trajectories of Fig. 2.3 raise another issue: the choice of X. Part of the optimal control problem formulation is the trajectory constraint $x(t) \in X$, i.e. the aim is to find the optimal trajectory from (x_0, q_0) (or the set of trajectories from O) to Γ that never leaves X. For control problems that do not experience state constraints of significant importance, the CDP Tool user still faces the problem of defining X. On one hand, a smaller X will lead to fewer discretization points and less computational complexity. On the other hand, to obtain the correct solution to the unconstrained problem, X must be large enough to include those trajectories. One practical approach to this problem is to start with a big X and a coarse gridding, make simulations based on the resulting value function to get a rough estimate of the trajectories, and then make a new choice of X based on this information.

Computing the Feedback Control Law

Provided that the lower bound, $V(x,q)$, is a good enough approximation of the optimal cost, the optimal feedback control law can be calculated as

$$\hat{u}(x,q) = \underset{u\in\Omega_u}{\operatorname{argmin}} \left\{ \frac{\partial V(x,q)}{\partial x} f(x,q,u) + l(x,q,u) \right\} \tag{2.15}$$

$$\hat{\mu}(x,q) = \underset{\mu\in\Omega_\mu}{\operatorname{argmin}} \{ V(x,\mu) + \hat{s}(x,q,\mu) \} \tag{2.16}$$

where $\hat{s}$ is identical to s with the exception that $s(x,q,\mu) = 0$ if $q = \mu$. The above expressions (implemented in `cdpctrl`) show a close resemblance to classical dynamic programming. Given an optimal value function V of an optimal control problem, (2.15) is how to derive the control signal for a purely continuous system, and (2.16) is how to derive the control signal for a purely discrete problem.

Simulation

The simulation commands take a hybrid system with a cost function and the associated control law as input and return the resulting trajectories, $x(t)$, $q(t)$, $u(t)$, and $J(t)$ or $J_e(t)$. The basic functions for simulations are `cdpsim` and `cdpsime`, but there also exist faster, less accurate fixed time step versions, `cdpsimf` and `cdpsimef`.

2.4 Examples

To further illustrate the usage of CDP Tool, this section shows the essential MATLAB code behind the the examples in [Hedlund and Rantzer, 1999].

EXAMPLE 2.1—A CAR WITH TWO GEARS
Consider the system

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= g_q(x_2)u, \quad q = 1,2 \quad |u| \leq 1 \end{cases} \tag{2.17}$$

where $g_q(x)$ is plotted in Fig. 2.4. This could be seen as a crude model of a car, u being the throttle, $g_q(x)$ the efficiency for gear number q.

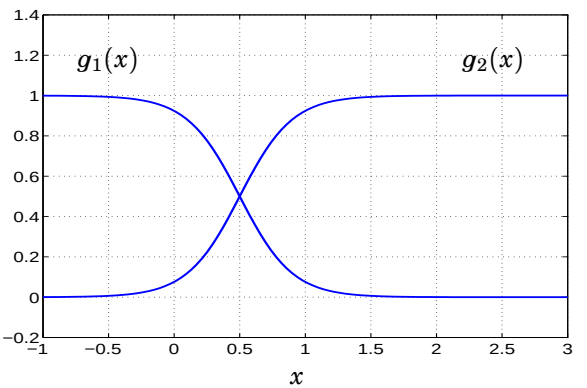


Figure 2.4 Gear efficiency at various speeds.

The problem is to bring (2.17) from $x_i = (-5, 0)$, $q_i = 1$ to $x_f = (0, 0)$, $q_f = 1$ in minimum time. Torque losses when using the clutch calls for an additional penalty for gear changes. Thus, the components of (2.3) have been chosen as $l_1(x, u) = l_2(x, u) = 1$, $s(x, 1, 2) = s(x, 2, 1) = 0.5$.

We start by writing the functions that define the system: f is entered into the file `car_f.m`, l into `car_l.m`, and finally s into `car_s.m`. We will not compute a true lower bound in this example. The extremal computations that would be needed for true bound purposes are included in the files anyway, to show how this could be done.

car_f.m

```
function y = car_f(x,q,u,h,vmode);
if (nargin > 3)
                        %%% perform extremal computations %%%
  minx2 = x(2)-h(2);      % min value of x2 over a square
  maxx2 = x(2)+h(2);      % max value of x2 over a square
  if (q==1)
    if (vmode == -1)      % component-wise minimization
      y = [minx2; sigmf(maxx2, [-5, 0.5])*u];
    elseif (vmode == 1)   % component-wise maximization
      y = [maxx2; sigmf(minx2, [-5, 0.5])*u];
    end;
  elseif(q==2)
    if (vmode == -1)      % component-wise minimization
      y = [minx2; sigmf(minx2, [5, 0.5])*u];
    elseif(vmode == 1)    % component-wise maximization
      y = [maxx2; sigmf(maxx2, [5, 0.5])*u];
    end;
  end;
else
                        %%% use the nominal value %%%
  if (q==1)
    y = [x(2); sigmf(x(2), [-5, 0.5])*u];
  elseif(q==2)
    y = [x(2); sigmf(x(2), [5, 0.5])*u];
  end;
end;
```

The function `sigmf` above gives

$$g_1(x_2) = \frac{1}{1+e^{5(x_2-0.5)}} \qquad \text{and} \qquad g_2(x_2) = \frac{1}{1+e^{-5(x_2-0.5)}}.$$

car_l.m

```
function y = car_l(x,q,u,h);
% For this example, l is the same regardless of the input.
na = nargin;    % dummy-line to allow variable number of inputs
y  = 1;
```

car_s.m

```
function y = car_s(x, q1, q2);
y = (q1~=q2)*0.5;            % The cost for switching is 0.5
```

Having entered these functions, we are ready to call `cdplowm` to get an approximation of the value function. Note that this is a minimum time problem that will lead to bang-bang control and Ω_u can be chosen as $\Omega_u = \{-1, 1\}$.

```
>> N    = [53;41];          % Number of grid points

>> xmin = [-6.5; -1.5];     % Define the state space
>> xmax = [5; 5.5];
>> Q    = 2;
>> XQ   = [xmin; xmax; Q];

>> uv   = [-1; 1];          % Define the control signal domain

>> xf   = [0;0];            % Gamma is a singleton
>> qf   = 1;
>> xqf  = [xf; qf];

>> omin = [-5.5; -0.5];     % define the optimization region
>> omax = [1; 3.0];
>> O    = [omin; omax];

>> [V,xv]   = cdplowm('car_f','car_l','car_s',uv,O,xqf,XQ,N,0);
```

Plots of the value function are shown in Figure 2.5 and 2.6 where x_i and x_f also have been marked. The functions look rather similar, since the cost for changing gears is only 0.5. One can see that V_1 has a threshold along the line $x_2 = 1$. Figure 2.4 reveals that the first gear is almost useless for high speeds, leading to $V_1 = V_2 + 0.5$ for $x_2 > 1$. This is the cost for using the second gear optimally after a gear switch.

We also compute a control law and use it in simulations

```
>> [U,Q] = cdpctrl('car_f','car_l','car_s',uv,Vc,xvc);

>> x0   = [-5;0];
>> q0   = 1;
>> xq0  = [x0; q0];
>> tend = 8;
>> [tv,xv2,qv] = cdpsim('car_f','car_l','car_s',U,Q,xvc,xq0,...
     [0;tend],xqf);
```

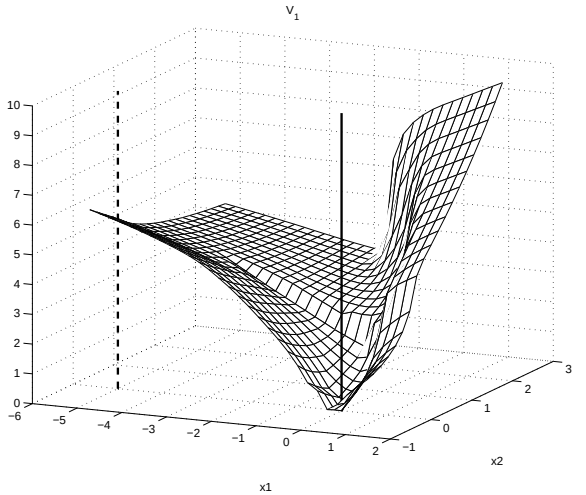


Figure 2.5 Plot of V_1. The initial point, x_i, is marked with a vertical dashed line, the final point, x_f, with a solid line.

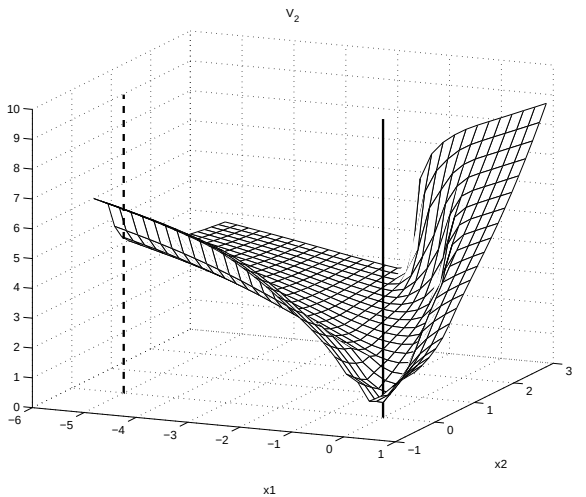


Figure 2.6 Plot of V_2.

The resulting trajectory is shown in Fig. 2.7, where the initial point is marked with a square. In the beginning, maximum throttle is used on the first gear (solid line). When the speed roughly reaches the point of equal efficiency between the gears ($x_2 = 0.5$), they are switched in favor of the

second gear (dashed line). At half the distance, the gas pedal is lightened to use the breaking force of the engine. In the end, the first gear is used again before the origin is hit. As seen in the figure, the granularity of the discretization grid ($h_1 = 0.22$, $h_2 = 0.18$) prevents the solution from hitting the exact origin.

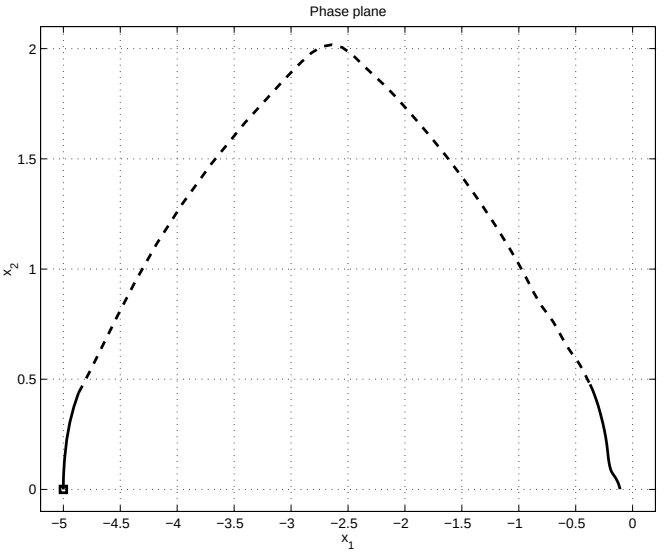


Figure 2.7 Phase portrait of a simulation. The solid line shows where gear number one has been used, the dashed line shows the second gear. The initial point is marked with a square.

Information about the optimal trajectory can also be found in the dual variables. The following code makes a single point maximization and extracts trajectory information from the dual variables of the solution.

```
>> [Vs,xvs,Ws] = cdplows('car_f','car_l','car_s',uv,xq0,...
     xqf,XQ,N,0);
>> Nxmin       = [-6; -0.5];
>> Nxmax       = [1; 3];
>> NX          = [Nxmin; Nxmax];
```

The optimal trajectory is easily found in Figs. 2.8 and 2.9, where the dual variables are plotted. The function $W_1(x)$ corresponds to the constraint (2.5) for $x \in X$, $q = 1$ and thus shows when the first gear has been used. Similarly, $W_2(x)$ corresponds to $q = 2$ and shows when the second gear has been used.

□

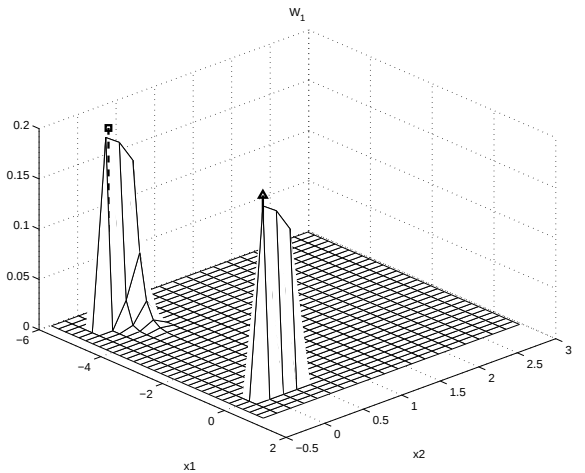


Figure 2.8 Plot of W_1. The initial point, x_i, is marked with a vertical dashed line, the final point, x_f, with a solid line.

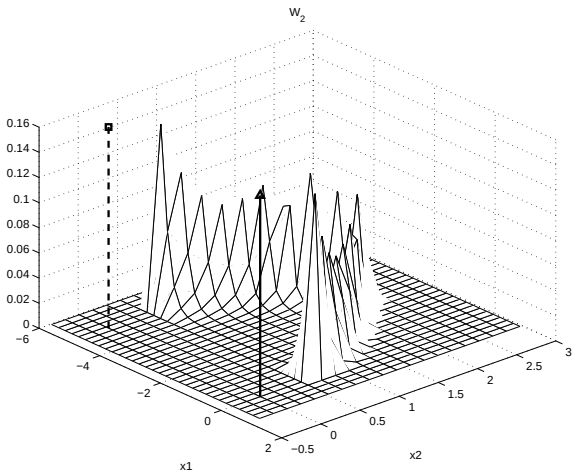


Figure 2.9 Plot of W_2.

EXAMPLE 2.2—ALTERNATE HEATING OF TWO FURNACES

Since the industrial power fee is determined by the highest peak of the season, it is desirable to spread the power consumption evenly over time. This is handled by load control, which means that the available electrical power is altered between different loads of the mill.

In this example, the temperature of two furnaces should be controlled

by alternate heating. The system has two continuous states that correspond to the temperature of the furnaces and is given by $\dot{x} = f_q(x)$, where

$$f_1(x) = \begin{bmatrix} -x_1 + u_0 \\ -2x_2 \end{bmatrix} \quad f_2(x) = \begin{bmatrix} -x_1 \\ -2x_2 + u_0 \end{bmatrix}$$
$$f_3(x) = \begin{bmatrix} -x_1 \\ -2x_2 \end{bmatrix}$$

Thus, there are three discrete modes: $q = 1$ means that the first furnace is heated, $q = 2$ means that the second furnace is heated, $q = 3$ corresponds to no heating. The cost function to be minimized is

$$J(x_0, q_0) = \int_{t_0}^{\infty} \sum_{i=1}^{2} (x_i - c_i)^2 e^{-t} dt + \sum_{k=1}^{M} b e^{-t_k}$$

where the desired stationary temperature values are $c_1 = 1/4$, $c_2 = 1/8$ and the cost for switching the power is $b = 1/1000$. Since the furnaces can only be fed by a fixed amount of energy, u_0, it is impossible to keep them stationary at the desired temperature. Hence, the time weighting, e^{-t}, is necessary to get a bounded cost function.

We start by writing the functions that define the system: f is entered into the file `furnace_f.m`, l into `furnace_l.m`, and finally s into `furnace_s.m`.

furnace_f.m

```
function y = furnace_f(x,q,u)
u0 = 0.8; % 0.8 will make the system enter mode 3 sometimes,
          % 0.4 prevents it
switch (q)
case 1,                          % Heating furnace no. 1
  y = [-x(1)+u0; -2*x(2)];
case 2,                          % Heating furnace no. 2
  y = [-x(1); -2*x(2)+u0];
case 3,                          % No heating
  y = [-x(1); -2*x(2)];
end;
```

furnace_l.m

```
function y = furnace_l(x,q,u)
y = (x(1)-0.25)^2+(x(2)-0.125)^2;
```

furnace_s.m

```
function y = furnace_s(x, q1, q2);
y = (q1~=q2)*0.001;            % Cost for switching: 0.001
```

With these functions, we are ready to call `cdplowem`. Note that we have no continuous input, u, in this example.

```
>> N     = [21; 21];

>> xmin  = [-0.10; -0.10];
>> xmax  = [0.50; 0.30];
>> Q     = 3;
>> XQ    = [xmin; xmax; Q];

>> uv    = [];

>> xf    = [0.25; 0.125];
>> qf    = 3;
>> xqf   = [xf; qf];

>> omin  = [-0.05; -0.05];
>> omax  = [0.40; 0.20];
>> O     = [omin; omax];

>> [V, xv]   = cdplowem('furnace_f','furnace_l','furnace_s',...
     1,uv,O,XQ,N,0);
```

The control law is derived and simulation is performed by calling `cdpsime` this time

```
>> [Um, Qm] = cdpctrl('furnace_f','furnace_l','furnace_s',...
     uv,Vc,xvc);

>> x0   = [0; 0];
>> q0   = 3;
>> xq0  = [x0; q0];
>> tend = 6;
>> [tv,xv2,qv] = cdpsime('furnace_f','furnace_l',...
     'furnace_s',1,Um,Qm,xvc,xq0,[0;tend]);
```

and the result is plotted in Fig. 2.10, which shows a time plot of the states and Fig. 2.11, which shows a phase portrait. The figures clearly show how the temperature of one furnace always decreases as the other one is heated. By alternate heating, the temperatures first climb up to, and above the set-point and then both furnaces are turned off and the

state drifts towards the origin. This procedure is then repeated over and over again, making the trajectory enclose the desired steady state (marked with a circle in the phase portrait). The trajectory has been dashed for $t \in [0, 3.5]$ in Fig. 2.11 to make the limit cycle clear.

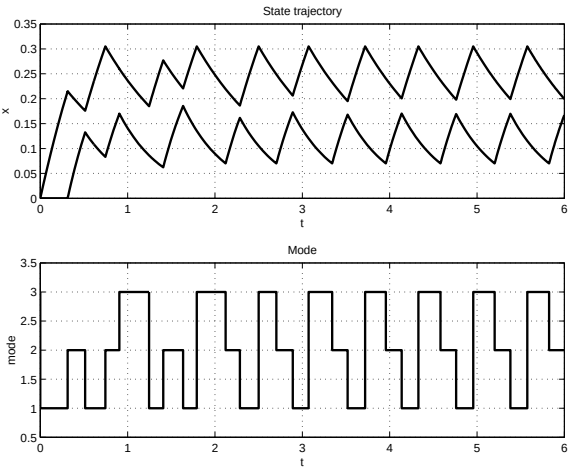


Figure 2.10 Time plot of the trajectories in the furnace example.

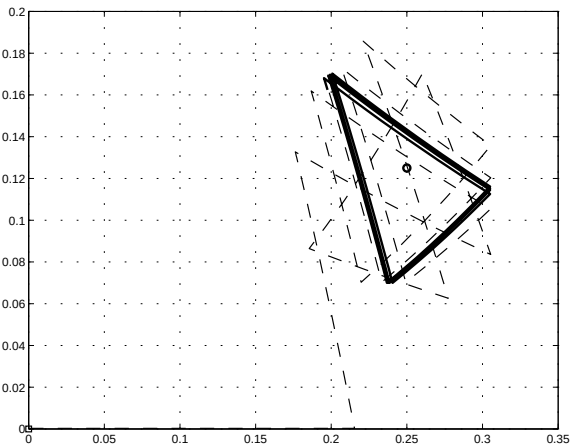


Figure 2.11 Phase portrait of the trajectories in the furnace example.

□

3

Case Study: CDP Tool in the Control Laboratory

3.1 Background

The basic control course is the first of the courses that the Automatic Control Department in Lund offers to Master students. This course, that teaches basic principles of automatic control, aims to give some insight to what is possible to achieve by control, as well as what is not possible. Linear continuous systems are treated.

The basic course contains three laboratory exercises, two of which use a double tank system to demonstrate PID control. The tank system, see Fig. 3.1, consists of a pump that is used to fill a tank. There is a hole in the bottom of the tank, with a second tank below. The water that flows out of the first tank, flows into the second tank. The water is drained from the second tank through a hole at the bottom similar to the hole in the first tank. Each tank has a level sensor.

A computer connected to the process contains a software PID controller that can use either of the level sensors as input. The controller output is the voltage to the pump. The main challenge during the laboratory exercise is to find controller parameters that give a desired step response for the tank levels.

In the second of the two laboratory exercises about PID control, the design method is pole placement. In the first part of the lab, the students make a model of the double tank system. First principles modeling leads

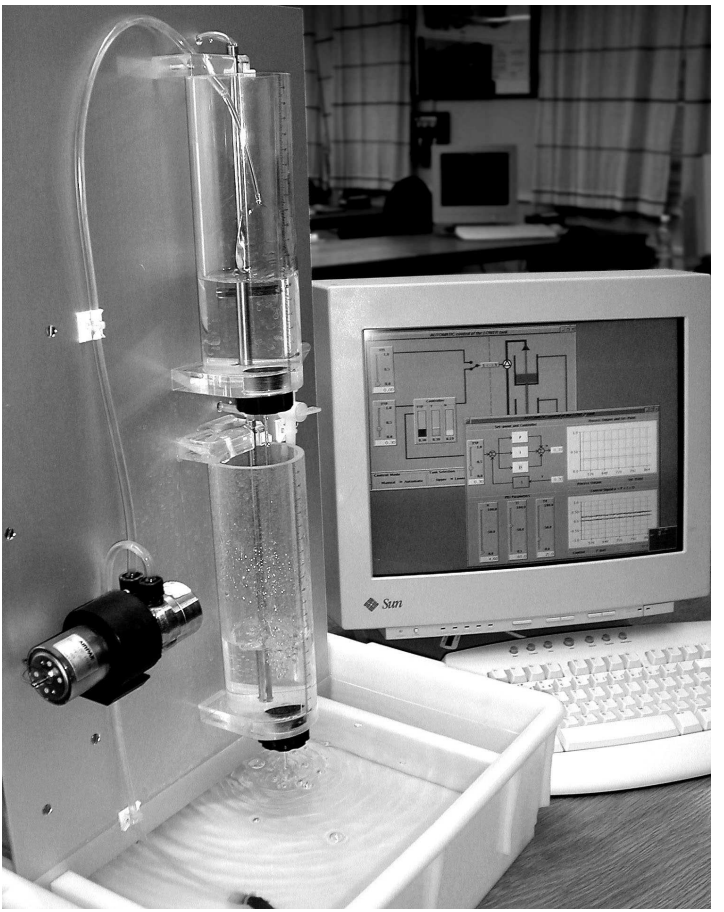


Figure 3.1 Double tank system and computer with software running

to a nonlinear state space model

$$\begin{aligned} \frac{dx_1(t)}{dt} &= -\alpha_1\sqrt{2gx_1(t)} + \beta u(t) \\ \frac{dx_2(t)}{dt} &= \alpha_1\sqrt{2gx_1(t)} - \alpha_2\sqrt{2gx_2(t)} \end{aligned} \tag{3.1}$$

where the two states x_1 and x_2 are normalized levels of the upper and the lower tank respectively, such that $0 \leq x_i \leq 1$, $i \in \{1, 2\}$. The value $x_i = 0$ corresponds to an empty tank and $x_i = 1$ means that the tank is full. The input, u, $0 \leq u \leq 1$, is normalized voltage to the pump.

The model is linearized and via Laplace transformed to a transfer function of the process, where the output is the level of the lower tank. Fast dynamics of the pump and the level sensor are neglected, leading to a second order system,

$$G_P(s) = \frac{K_p T_2}{(1+sT_1)(1+sT_2)} \tag{3.2}$$

where T_1 and T_2 are the time constants of the two tanks, and K_p is a gain constant.

The lower tank should be controlled using a PID controller

$$G_R(s) = K\left(1 + \frac{1}{sT_i} + sT_d\right)$$

where K, T_i, and T_d are the controller parameters to be determined.

The resulting closed loop system is of third order

$$\begin{aligned} G_{cl}(s) &= \frac{G_P(s)G_R(s)}{1+G_P(s)G_R(s)} \\ &= \frac{K_p K T_2(s^2 T_i T_d + sT_i + 1)}{s^3 T_i T_1 T_2 + s^2 T_i(T_1 + T_2 + K_p K T_2 T_d) + sT_i(1 + K_p K T_2) + K_p K T_2} \end{aligned}$$

and the three controller parameters can be used to adjust the closed loop dynamics. The desired closed loop system is characterized in terms of speed, ω, and relative damping ζ via the requested characteristic equation $(s+\omega)(s^2 + 2\zeta\omega s + \omega^2)$.

This leads to the following (approximate) map from the desired closed loop behavior, ω and ζ to the controller parameters

$$\begin{aligned} K &= \frac{T_1\omega^2}{K_p}(1+2\zeta) \\ T_i &= \frac{1+2\zeta}{\omega} \\ T_d &= \frac{1+2\zeta}{\omega(1+2\zeta)} \end{aligned}$$

which is used to examine the lab process for different values of ω and ζ.

One purpose of this laboratory exercise is to show the students how the theory can work in practice. Another lesson to learn, is to avoid relying on assumptions that may be invalid. Real life offers properties that the model in Eq. (3.2) does not capture, e.g. nonlinearities such as the limits of the control signal.

3.2 Practical Problems During the Exercise

During the exercise, the closed loop system is thoroughly examined by repeated step responses in the lower tank level, making series of experiments with systematic changes in either speed or damping of the poles. Many of the tested controllers do not perform very well (either by request of slow or poorly damped response in the exercise instruction or by miscalculation by the student). For consistent experiments, all step responses should start at the same initial state. If the chosen controller parameters lead to a very long settling time of the step response, the preparatory step back to the initial state before the next iteration will involve long waiting time for the students. In addition to this annoyance, bad choice of parameters often lead to flooding of the upper tank and the lab risks being experienced as tedious and splashy.

3.3 Solution

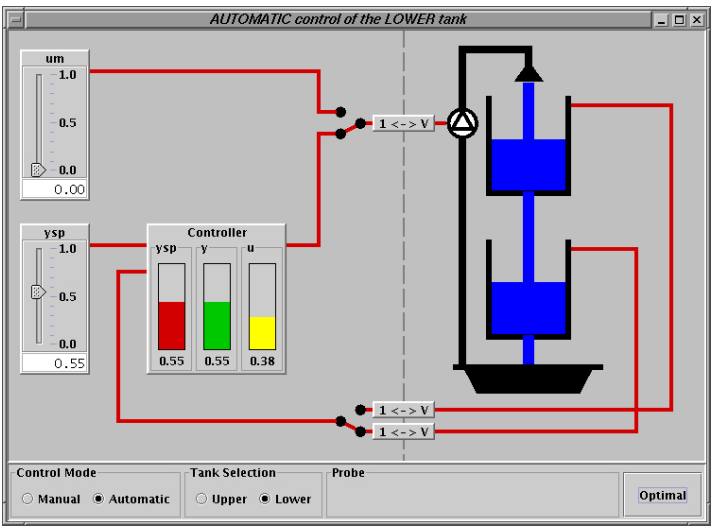


Figure 3.2 The process view of the double tank controller interface.

The introduction of new software and interface for the controller in 2002 made it possible to also include new functionality designed using CDP Tool. A feature of the new interface is the “Optimal” button. When clicking this button, the current PID controller is temporarily discarded and

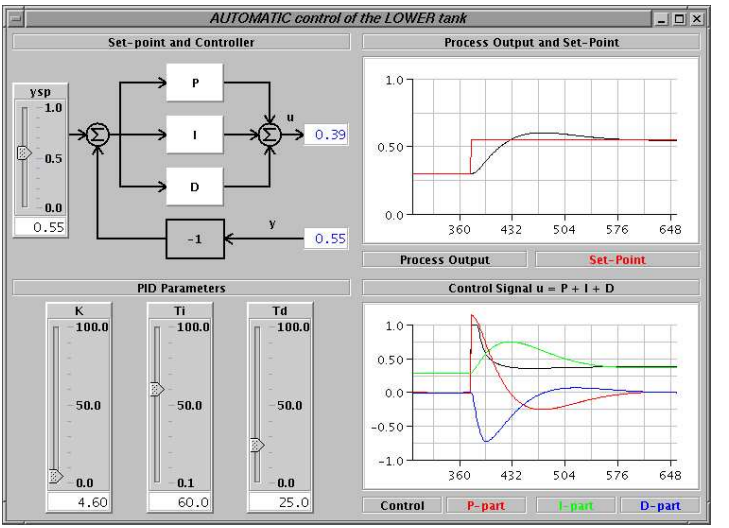


Figure 3.3 The controller view of the double tank controller interface.

replaced by an optimal controller that brings both tank levels to the current set point in minimum time. When the process has reached the set point, given by the sliders labeled "ysp" in Figs. 3.2 and 3.3, the optimal controller hands over to the PID controller again via a bumpless transfer. This functionality reduces the idle time between step response experiments and quickly resolves situations of tank flooding and seemingly never-ending oscillations.

The following section shows how the optimal controller was computed.

3.4 CDP Tool Design Parameters

Before discussing how to find the parameters, recall the problem that CDP Tool tries to solve (pure continuous system version):

Given the state space $X \subset \mathbf{R}^n$, the final set $\Gamma \subset X$, and a control feedback law $u = u(x)$, $x \in X$, define $V_u : X \to \mathbf{R}$ as

$$V_u(x_0) = \int_0^T l\Big(x(t), u\big(x(t)\big)\Big)dt \tag{3.3}$$

subject to

$$\dot{x}(t) = f\Big(x(t), u\big(x(t)\big)\Big) \tag{3.4}$$

$$x(t) \in X, \quad 0 \le t \le T \tag{3.5}$$

$$x(0) = x_0 \tag{3.6}$$

$$x(T) \in \Gamma \tag{3.7}$$

The CDP Tool optimal control problem is to find $V^*(x_0) = \min_u V_u(x_0)$ for each $x_0 \in O \subset X$, where O is the optimization region.

For the double tank process, the states are the water levels, i.e. $x = (x_1, x_2)$, the function $f(x,u)$ is given by Eq. (3.1), and $l(x,u) = 1$ for minimum time control.

Two sets that should be fed to CDP Tool are the trajectory space, X, and the optimization region, $O \subset X$. The optimization region is the part of the state space for which the value function and the feedback control law should be computed. The trajectory space is the state space where trajectories are allowed.

Since $x_i = 0$, $i = \{1,2\}$ means that tank i is empty, and $x_i = 1$ means that the tank is full, $0 \le x_i(t) \le 1$ is a hard constraint on the trajectories. The control feedback law will be needed for all states within these constraints, so a natural choice is $X = O = [0,1] \times [0,1]$. Unwanted asymmetries for the discretization scheme at the boundaries of X, however, makes it desirable to let O be strictly contained in X. Thus, let

$$O = [0,1] \times [0,1] \text{ and} \tag{3.8}$$

$$X = [-0.05, 1.05] \times [-0.05, 1.05]. \tag{3.9}$$

To overcome numerical problems from very high gain of the system close to $x = (0,0)$ and to extend the model down to $x_i = -0.05$, the square root function $\sqrt{2gx_i}$ is replaced by its linearization below a certain level, $x^0 = 0.05$:

$$\sqrt{2gx^0} + \sqrt{\frac{g}{2x^0}}(x_i - x^0) \tag{3.10}$$

To account for variations among the level sensors and to avoid flooding the tanks, it is desirable to keep the levels below 0.95. For low levels, the measurements are very noisy because of slosh and the model (3.1) is invalid (it would be invalid even without the modification of the square root function, since Torricelli's law does not account for turbulence), so the aim is to keep the levels above 0.05.

The way to do this is to modify $l(x,u)$ to give a high penalty for undesirable states:

$$l(x,u) = \begin{cases} 1, & 0.05 \leq x_1 \leq 0.95,\ 0.05 \leq x_2 \leq 0.95 \\ l_{\text{high}}, & \text{otherwise} \end{cases} \tag{3.11}$$

where l_{high} is a constant $\gg 1$, e.g. $l_{\text{high}} = 1000$.

Among the design choices is also the granularity of the discretization grid. For this example, 50 points along each coordinate axis of X is sufficient. Knowing (via the Pontryagin Maximum Principle) that time optimal control of a system that is affine in the control signal leads to a bang-bang solution, u can be discretized as $u \in \{0,1\}$.

3.5 Implementation

Since the optimal control problem is computationally demanding and too complex to solve online, CDP Tool is used to compute control signal lookup tables that can be stored in a text file and fed to the Java based controller. Given the problem, CDP Tool first computes the value function and then the control law. The control law is given as the value of u for each grid point in O. Multilinear interpolation can then be used to get an approximate value in between the grid points.

The control problem in Eqs. (3.3)–(3.7) is defined for a fixed target set, Γ. To handle arbitrary setpoints, the CDP Tool must be used several times, once for each desired setpoint. For the double tank example, eighteen iterations are used, with Γ set to $\{(0.05k, 0.05k)\}$ under the k:th iteration. The final outcome of all CDP Tool computations is thus a three dimensional lookup table that returns the control signal for different values of x_1, x_2, and the setpoint.

3.6 Benefits of the “Optimal”-button

Fig. 3.4 shows a comparison between a well tuned PID controller and the time optimal controller. Well tuned means that the settling time of the step response is as short as possible without flooding the upper tank. The dashed curve and the solid curve are the levels in the upper and lower tank respectively. The thin solid line is the reference value. The first 700 seconds show a step response forth and back using the PID controller, while the remaining time is a step response resulting from the time optimal controller.

Fig. 3.4 also shows some robustness to process variations in the optimal controller. The controller has been designed using a model where the two tank levels are equal in stationarity. It is seen in the figure that the controller works well, though the stationary levels differ significantly around 0.55.

The PID controller is slower than the optimal controller and this difference is particularly noticeable in the step downwards. It is not possible to reverse the pump to empty the tank, but the time optimal approach is to temporarily lower the level of the upper tank significantly below its final value, to be able to drain the lower tank faster.

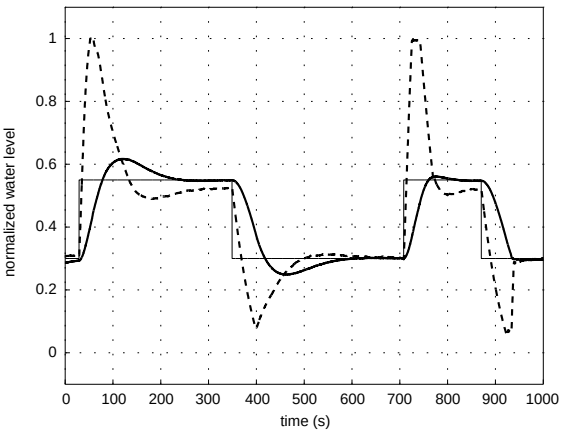


Figure 3.4 A performance comparison between a well tuned PID controller and a time optimal controller. The dashed line is the upper tank level, the solid line is the lower tank level. The PID controller is used the first 700 seconds, the optimal controller is used the last 300 seconds.

It is of course not fair to compare these two controllers. The PID controller on one hand, is linear with limited tuning possibilities, while the optimal controller, on the other hand, is model based and takes care of state constraints, input constraints, using the nonlinear model. The plot serves, however, as an illustration of the benefits of using the optimal controller in the lab.

In addition to speed up parts of the laboratory exercise, the optimal controller also works as “panic resolver”. Controller parameters that lead to a poorly damped system, will give long lasting transients and sometimes flooding from a step response. The “Optimal” button will then quickly bring the system back to rest again.

Today the laboratory exercise requires less idle time for the students, leading to less boredom. Some of the students get fascinated and interested in how this magic button does its trick. The obvious salesman's answer is "You can learn about optimal control in our optional, advanced courses".

4

Bibliography

Alur, R. and D. L. Dill (1994): "A theory of timed automata." *Theoretical Computer Science*, **126:2**, pp. 183–235.

Alur, R., T. A. Henzinger, and H. Wong-Toi (1997): "Symbolic analysis of hybrid systems." In *Proceedings of the 36th IEEE Conference on Decision and Control*, pp. 702–707.

Antsaklis, P. J. and A. Nerode (1998): "Hybrid control systems: An introductory discussion to the special issue." *IEEE Transactions on Automatic Control*, **43:4**, pp. 457–460. Special issue on hybrid systems.

Årzén, K.-E., M. Johansson, and R. Babuska (1998): "A survey on fuzzy control." Technical Report. Esprit LTR project FAMIMO Deliverable D1.1B.

Asarin, E., O. Bournez, T. Dang, and O. Maler (2000): "Approximate reachability analysis of piecewise-linear dynamical systems." In *Lecture Notes in Computer Science. Hybrid Systems: Computation and Control*, pp. 21–31. Springer-Verlag, Heidelberg, Germany.

Bardi, M. and I. Capuzzo-Dolcetta (1997): *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Birkhauser, Boston.

Bellman, R. (1957): *Dynamic Programming*. Princeton University Press.

Bemporad, A. and M. Morari (1999a): "Control of systems integrating logic, dynamics, and constraints." *Automatica*, **35:3**, pp. 407–427.

Bemporad, A. and M. Morari (1999b): "Verification of hybrid systems via mathematical programming." In *Hybrid Systems: Computation and Control; Second International Workshop*. Nijmegen, The Netherlands.

Bensoussan, A. and J. L. Menaldi (1997): "Hybrid control and dynamic programming." *Dynamics of Continuous, Discrete and Impulsive Systems*, **3:4**, pp. 395–442.

Bertsekas, D. P. and J. N. Tsitsiklis (1996): *Neuro-dynamic programming*. Athena Scientific.

Branicky, M. (1998): "Multiple Lyapunov functions and other analysis tools for switched and hybrid systems." *IEEE Transactions on Automatic Control*, **43:4**, pp. 475–482. Special issue on hybrid systems.

Branicky, M. S., V. S. Borkar, and S. K. Mitter (1994): "A unified framework for hybrid control." In *Proceedings of the 33rd Conference on Decision and Control*, pp. 4228–4234.

Branicky, M. S. and S. K. Mitter (1995): "Algorithms for optimal hybrid control." In *Proceedings of the 34th Conference on Decision & Control*. New Orleans.

Corless, M. (1994): "Robust stability analysis and controller design with quadratic lyapunov functions." In Zinober, Ed., *Variable Structure and Lyapunov Control*, chapter 9, pp. 181–203. Springer Verlag.

Cormen, T. H., C. E. Leiserson, and R. L. Rivest (1989): *Introduction to Algorithms*. MIT Press.

Daws, C., A. Olivero, and S. Yovine (1994): "Verifying ET-LOTOS programs with KRONOS." In *Proc. 7th FORTE'94*, pp. 227–242. Chapman & Hall.

Dunford, N. and J. T. Schwartz (1958): *Linear Operators, Part I: General Threory*. Interscience Publishers, Inc.

Evans, L. C. (1997): "Partial differential equations and Monge-Kantorovich mass transfer.". Department of Mathematics, University of California, Berkeley.

Ezzine, J. and A. H. Haddad (1989): "Controllability and observability of hybrid systems." *Int. J. Contr.*, **49**, June, pp. 2045–2055.

Filippov, A. F. (1988): *Differential Equations with Discontinuous Right-hand Sides*. Kluwer Academic Publishers.

Ford, L. K. and D. K. Fulkerson (1962): *Flows in Networks*. Princeton University Press, Princeton, New Jersey.

Göllü, A. and P. Varaiya (1989): "Hybrid dynamical systems." In *Proceedings of the 28th IEEE Conference on Decision and Control*, pp. 2708–2712.

Grammel, G. (1999): "Maximum principle for a hybrid system via singular perturbations." *SIAM Journal of Control and Optimization*, **37:4**, pp. 1162–1175.

Hedlund, S. (1999): "Computational methods for hybrid systems." Technical Report Licentiate thesis ISRN LUTFD2/TFRT--3225--SE. Department of Automatic Control, Lund Institute of Technology, Sweden.

Hedlund, S. and M. Johansson (1999): "PWLTool — A Matlab toolbox for analysis of piecewise linear systems." Report TFRT-7582. Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Hedlund, S. and A. Rantzer (1999): "Optimal control of hybrid systems." In *Proceedings 38th IEEE Conference on Decision and Control*. Phoenix, Arizona.

Hedlund, S. and A. Rantzer (2000): "Hybrid control laws from convex dynamic programming." In *IEEE Conference on Decision and Control*. Sydney.

Hedlund, S. and A. Rantzer (2002): "Convex dynamic programming for hybrid systems." *IEEE Transactions on Automatic Control*, **47:9**, pp. 1536–1540.

Heemels, W. P. M. H. (1999): *Linear Complementarity Systems*. PhD thesis, Eindhoven University of Technology.

Heemels, W. P. M. H., B. D. Schutter, and A. Bemporad (2001): "Equivalence of hybrid dynamical models." *Automatica*, **37:2**, pp. 1085–1091.

Henzinger, T. A., P.-H. Ho, and H. Wong-Toi (1995): "A user guide to HyTech." In Springer-Verlag, Ed., *Proceedings of the First International Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 41–71.

Henzinger, T. A., P. W. Kopke, A. Puri, and P. Varaiya (1998): "What's decidable about hybrid automata?" *Journal of Computer and System Sciences*.

Hespanha, J. P., D. Liberzon, and E. D. Sontag (2002): "Nonlinear observability and an invariance principle for switched systems." In *Proceedings of the 41st IEEE Conference on Decision and Control*, pp. 4300–4305. Las Vegas, Nevada USA.

Hitchcock, F. L. (1941): "The distribution of a product from several sources to numerous localities." *J. Math. Phys.*, **20**, pp. 224–230.

Hou, L., A. N. Michel, and H. Yie (1996): "Stability analysis of switched systems." In *Proceedings of the 35th Conference of Decision and Control*, pp. 1208–1212. Kobe, Japan.

IEEE (1998): "Special issue on hybrid systems." *IEEE Transactions on Automatic Control*, **43**, April.

IEEE (2000): "Special issue on hybrid systems: Theory and applications." *Proceedings of the IEEE*, **88**, July.

Joahnsson, M. and A. Rantzer (1996): "Computation of piecewise quadratic Lyapunov functions for hybrid systems." Technical Report ISRN LUTFD2/TFRT--7459--SE. Department of Automatic Control, Lund Institute of Technology, Sweden.

Johansson, K. H. (1997): *Relay Feedback and Multivariable Control*. PhD thesis ISRN LUTFD2/TFRT--1048--SE, Department of Automatic Control, Lund Institute of Technology, Sweden.

Johansson, K. H., J. Lygeros, and S. Sastry (2003): *Encyclopedia of Life Support Systems*, chapter 6.43.28.1 Modeling of Hybrid Systems. UNESCO-EOLSS Joint Committee. To be published.

Johansson, M. (1999): *Piecewise Linear Control Systems*. PhD thesis ISRN LUTFD2/TFRT--1052--SE, Department of Automatic Control, Lund Institute of Technology, Sweden.

Johansson, M. (2002): *Piecewise Linear Control Systems*. Springer-Verlag, Heidelberg, Germany.

Kantorovich, L. (1942): "On the transfer of masses." *Dokl. Akad. Nauk. SSSR*, **37**, pp. 227–229. In russian.

Koopmans, T. C. (1947): "Optimum utilization of the transportation system." In *Proceedings of the international statistical conference*. Washington, D.C.

Larsen, K. G., P. Pettersson, and W. Yi (1995): "Compositional and symbolic model-checking of real-time systems." In *Proceedings of the 16th IEEE Real-Time Systems Symposium*, pp. 76–87.

Luenberger, D. G. (1969): *Optimization by Vector Space Methods*. Wiley.

Lygeros, J., K. Johansson, S. Sastry, and M. Egerstedt (1999): "On the existence of executions of hybrid automata." In *Proceedings of the 38th IEEE Conference on Decision and Control*, pp. 2249–2254.

Pepyne, D. L. and C. G. Cassandras (2000): "Optimal control of hybrid systems in manufacturing." *Proceedings of the IEEE*, **88:7**, pp. 1108–1123.

Piccoli, B. (1999): "Necessary conditions for hybrid optimization." In *Proceedings of the 38th IEEE Conference on Decision and Control*, pp. 410–415.

Pontryagin, L. S., V. Boltyansky, R. Gamkrelidze, and E. Mishchenko (1962): *The Mathematical Theory of Optimal Processes*. Wiley.

Rachev, S. and L. Rüschendorf (1998): *Mass Transporation Problems, Volume I: Theory*. Probability and its Applications. Springer.

Rantzer, A. (1999): "Dynamic programming via convex optimization." In *Proceedings of the IFAC world congress*. Beijing.

Rantzer, A. (2001): "A dual to Lyapunov's stability theorem." *Systems & Control Letters*, **42:3**, pp. 161–168.

Rantzer, A. and F. Ceragioli (2001): "Smooth blending of nonlinear controllers using density functions." In *Proceedings of European Control Conference*.

Rantzer, A. and M. Johansson (2000): "Piecewise linear quadratic optimal control." *IEEE Transactions on Automatic Control*, April.

Rantzer, A. and P. Parrilo (2000): "On convexity in stabilization of nonlinear systems." In *Proceedings of IEEE Conference of Decision and Control*.

Riedinger, P. and C. Iung (1999): "Optimal control for hybrid systems: An hysteresis example." In *IEEE International conference on Systems, Man, and Cybernetics*, vol. 1, pp. 188–193.

Riedinger, P., F. Kratz, C. Iung, and C. Zanne (1999): "Linear quadratic optimization for hybrid systems." In *Proceedings of the 38th IEEE Conference on Decision and Control*, pp. 3059–3064.

Rudin, W. (1991): *Functional Analysis*, 2nd edition. McGraw-Hill.

Schutter, B. D. and T. van den Boom (2001): "Model predictive control for max-min-plus-scaling systems." In *Proceedings of the 2001 American Control Conference*, pp. 319–324.

Silva, B. and B. Krogh. (2000): "Formal verification of hybrid systems using checkMate: A case study." In *American Control Conference*.

Sontag, E. D. (1981): "Nonlinear regulation: the piecewise linear approach." *IEEE Transactions on Automatic Control*, **26:2**, pp. 346–357.

Sontag, E. D. (1996): "Interconnected automata and linear systems: A theoretical framework in discrete time." In Alur *et al.*, Eds., *Hybrid Systems III: Verification and Control*, pp. 436–448. Springer, NY.

Stiver, J. A. and P. J. Antsaklis (1992): "Modeling and analysis of hybrid control systems." In *Proceedings of the 31st IEEE Conference on Decision and Control*, vol. 4, pp. 3748–3751.

Sussmann, H. J. (1999): "A maximum principle for hybrid optimal control problems." In *Proceedings of the 38th IEEE Conference on Decision and Control*, pp. 425–430.

Tomlin, C. J., J. Lygeros, and S. S. Sastry (2000): "A game theoretic approach to controller design for hybrid systems." *Proceedings of the IEEE*, **88:7**, pp. 949–970.

Utkin, V. I. (1977): "Variable structure systems with sliding modes." *IEEE Transactions on Automatic Control*, **AC-22**, pp. 212–222.

van der Schaft, A. J. and H. Schumacher (1998): "Complementarity modeling of hybrid systems." *IEEE Transactions on Automatic Control*, **43:4**, pp. 483–490.

van der Schaft, A. J. and H. Schumacher (2000): *An Introduction to Hybrid Dynamical Systems*. Springer.

Vinter, R. (1993): "Convex duality and nonlinear optimal control." *SIAM J. Control and optimization*, **31:2**, pp. 518–538.

Xu, X. and P. J. Antsaklis (2000): "Optimal control of switched systems: New results and open problems." In *Proceedings of the American Control Conference*, pp. 2683–2687. Chicago, Illinois.

Young, L. C. (1969): *Lectures on the Calculus of Variations and Optimal Control Theory*. W. B. Saunders Company, Philadelphia, Pa.