Robustness, Adaptation, and Learning in Optimal Control

Thesis by

Ivan Papusha

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2016

(Defended May 12, 2016)

To my family.

Acknowledgments

First and foremost, I would like to thank my incredible advisor, Prof. Richard Murray, for his guidance and support during my graduate school years. He helped me clarify ideas, and gave me great freedom to pursue the best ones. Despite his full schedule, Richard always made time for me. He has forever earned a place in the back of my mind as the voice that asks questions like "why?" "what is the big picture?" and "suppose you could do this. . . "

Huge thanks also go to my thesis committee, Profs. John Doyle, Joel Burdick, and Eugene Lavretsky. Their great courses and sharp insight into robust control, robotics and estimation, and adaptation got me excited about the intersection between control theory and engineering. I was extremely privileged to have the geniuses of my field, right here at Caltech, helping to answer my questions and giving advice. I also want to acknowledge my officemates and close friends S. You, A. Swaminathan, K. Shan, as well as my peers and collaborators J. Fu, U. Topcu, E. Wolff, S. Livingston, I. Filippidis, K. Shankar, M. Horowitz, S. Farahani, C. McGhan, N. Matni, and V. Jonsson for many insightful discussions.

I want to give a special thanks to Prof. Vijay Kumar and his group at the University of Pennsylvania, especially M. Pivtoraiko and R. Ramaithitima, for graciously providing the facilities and expertise with the quadrotor experiments that appear in this thesis.

In addition, I wanted to acknowledge Profs. Stephen Boyd, Sanjay Lall, and Andrew Ng at Stanford. Their clear and precise teaching had a significant, lasting impact on my academic style, and their recommendation letters helped get me into graduate school. I also had a lot of fun TA-ing thanks to Prof. Doug MacMartin and his great knowledge of classical control. Other friends and colleagues from NorCal, SoCal, and everywhere else who helped shape this work and keep me motivated include V. Athalye, V. Gund, G. Schiebinger, D. Chen, D. Kim, K. Seetharam, J. Kindem et al., R. McVay et al., B. O'Donoghue, E. Chu, M. Kraning, E. Akuiyibo, Y. Wang, and J. Ngiam. I also want to acknowledge my coworkers at AOL, including A. Flores, J. Batiz, M. Agarwal, and V. Andrei.

Of course, I would not be anywhere without my amazing family. Great thanks go to my mom, grandma, and little sister Katya, for providing support and encouragement from across the continent.

Finally, I gratefully acknowledge support from the Department of Defense through the NDSEG Fellowship Program, the Boeing Company, the Powell Foundation, and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

Abstract

Recent technological advances have opened the door to a wide variety of dynamic control applications, which are enabled by increasing computational power in ever smaller devices. These advances are backed by reliable optimization algorithms that allow specification, synthesis, and embedded implementation of sophisticated learning-based controllers. However, as control systems become more pervasive, dynamic, and complex, the control algorithms governing them become more complex to design and analyze. In many cases, optimal control policies are practically impossible to determine unless the state dimension is small, or the dynamics are simple. Thus, in order to make implementation progress, the control designer must specialize to suboptimal architectures and approximate control. The major engineering challenge in the upcoming decades will be how to cope with the complexity of designing implementable control architectures for these smart systems while certifying their safety, robustness, and performance.

This thesis tackles the design and verification complexity by carefully employing tractable lower and upper bounds on the Lyapunov function, while making connections to robust control, formal synthesis, and machine learning. Specifically, optimization-based upper bounds are used to specify robust controllers, while lower bounds are used to obtain performance bounds and to synthesize approximately optimal policies. Implementation of these bounds depends critically on carrying out learning and optimization in the loop. Examples in aerospace, formal methods, hybrid systems, and networked adaptive systems are given, and novel sources of identifiability and persistence of excitation are discussed.

Contents

List of Symbols

Notation	Meaning
$\mathbf{R}$, $\mathbf{R}^n$, $\mathbf{R}^{m\times n}$	real numbers, n-dimensional vectors, and $m \times n$ matrices
$\mathbf{C}$, $\mathbf{C}^n$, $\mathbf{C}^{m\times n}$	complex numbers, n-dimensional vectors, and $m \times n$ matrices
j	the imaginary unit, $\sqrt{-1}$
$\mathbf{1}$	the vector of all ones, *i.e.*, $(1, \ldots, 1) \in \mathbf{R}^n$
$\mathbf{S}^n$	symmetric $n \times n$ matrices $\{X \in \mathbf{R}^{n\times n} \mid X = X^T\}$
$\mathbf{S}^n_+$	positive semidefinite matrices $\{X \in \mathbf{S}^n \mid z^T X z \geq 0 \text{ for all } z \neq 0\}$
$\mathbf{S}^n_{++}$	positive definite matrices $\{X \in \mathbf{S}^n \mid z^T X z > 0 \text{ for all } z \neq 0\}$
$P \succeq 0$, $P \succ 0$	same as $P \in \mathbf{S}^n_+$, $P \in \mathbf{S}^n_{++}$, dimension taken from context
$P \succeq_K Q$, $P \succ_K Q$	$P - Q \in \mathbf{cl}\, K$, and $P - Q \in \mathbf{int}\, K$, where K is a convex cone
$\mathbf{Tr}(M)$	trace of a matrix $M \in \mathbf{R}^{n\times n}$, *i.e.*, $\sum_{i=1}^n M_{ii}$
$\lambda_{\max}(P)$, $\lambda_{\min}(P)$	maximum, minimum eigenvalue of a matrix $P \in \mathbf{S}^n$
$\sigma_{\max}(M)$, $\sigma_{\min}(M)$	maximum, minimum singular value of a matrix $M \in \mathbf{R}^{m\times n}$
$\mathbf{Co}(S)$	convex hull of the set $S \subseteq \mathbf{R}^n$, given by $\mathbf{Co}(S) = \left\{ \sum_{i=1}^k \theta_i x_i \mid x_i \in S,\ \theta_i \geq 0, i = 1, \ldots, k,\ \sum_{i=1}^k \theta_i = 1 \right\}$
$\mathbf{L}_p$	signals $u : \mathbf{R}_+ \to \mathbf{C}^n$ with $\|u\|_p = \left(\int_0^\infty \|u(\tau)\|_p \, d\tau\right)^{1/p} < \infty$, $1 \leq p \leq \infty$
$\|u\|$	2-norm for vectors, $\mathbf{L}_2$-norm for signals, $\sigma_{\max}(u)$ for matrices
$\mathbf{H}_\infty$	Hardy space on $\mathbf{C}^n_+$
$\mathbf{E}\, x$	expected value of a random variable x
$\mathbf{P}(A)$	probability of an event A
$\hat{\theta}(t), \hat{y}(t)$	estimate at time t of a parameter $\theta \in \mathbf{R}^{n_\theta}$, and output $y \in \mathbf{R}^{n_y}$
$\tilde{\theta}(t), \Delta\theta(t)$	parameter estimation error $\hat{\theta}(t) - \theta \in \mathbf{R}^{n_\theta}$
$\tilde{y}(t), \Delta y(t)$	output prediction error $\hat{y}(t) - y(t) \in \mathbf{R}^{n_y}$

Chapter 1

Introduction

From its classical engineering roots in the 1930s–40s, increasing mathematization in the 1950s–60s, through to the present day, the discipline of control has been progressive in its adoption of computational tools and algorithms. The Nyquist criterion, Bode and Nichols plots, Kalman filter, KYP lemma, LMIs, and convex optimization are all tools designed by control theorists, or imported from other fields to solve an engineering question. But instead of giving a specific answer, more often than not these theoretical tools lead to a procedure, which the control designer—in their quest for practical implementation—must execute, check and recheck, abandon, resurrect, tune, and check once more. The final proof is in the pudding. After all, the goal is to control dynamical systems.

This reliance on algorithms starkly conflicts with the standard practice in mathematics and physics, where definite answers reign supreme. A true 17th century mathematician would never be satisfied until a closed-form solution to a problem has been obtained. Only a formula, or an equation, a concept—something simple—into which someone could substitute facts and figures, would make them happy. Whereas a mathematician wants to tell you *what* or *why* something is, a control theorist would be more interested in *how* to make it happen.

These algorithmic tools help researchers and decision makers working in a variety of multidisciplinary fields, such as biology and medicine, transportation, and earth sciences. By making use of a strong tradition in modeling and abstraction of physical systems, they are able to solve problems they could not before. Control theory has the right pedigree to help contextualize some of the toughest "squishy" and "applied" questions, because it lies at the intersection of engineering and mathematics, and has an obsession with both intellectual rigor and practicality.

At the same time, control theory is undergoing its own revolution driven by ever faster

computers, and ever better optimization algorithms. As our world becomes increasingly automatized, engineers are faced with very real societal needs to create complex and optimized automatic systems that affect more and more people. Meanwhile, these systems must operate safely, robustly, and with high performance. This means that practice continues to outpace rationale. In fact, the late Jan Willems predicted the influx of optimization technology as the next "big thing" in control. In his autobiographical essay, he wrote

> ... MPC is an area where essentially all aspects of the field, from modeling to optimal control, and from observers to identification and adaptation, are in synergy with computer control and numerical mathematics [WIL07].

MPC stands for Model Predictive Control, a particular kind of optimization-based control method that comes from the chemical process community. Its main idea is very simple:

1. model the system
2. using the model, make a plan from now until a time T later (with optimization)
3. execute the first portion of the plan, and let the system evolve
4. go to step 2.

Model Predictive Control is arguably one of the most compelling, highly performant, and most broadly applicable tools control theory has to offer. For example, it has the ability to incorporate constraints on the input and state. It neatly ties together observers and controllers. It can be extended to nonlinear, hybrid, stochastic, and machine learning systems. It has beautiful links to dynamic programming, and encompasses all of LQR, LQG, and Kalman filtering.

MPC was a great idea for chemical processes. Because chemical processes operate at slow timescales (≤ 1 Hz), optimal control calculations (step 2) could easily be performed between data points. For years, successful application of MPC to faster, more dynamic plants has been held back by slow computers and complex algorithms. But now that computers and optimization methods are catching up, MPC can finally be applied to these faster plants (10^1–10^5 Hz). The next few decades of research in system theory will be devoted to figuring out just how far optimization-based control can be taken in practice. Already, fast computers are opening up a huge variety of dynamically interesting applications in robotics, aerospace, finance, and transportation.

However, the very thing that makes control relevant, namely *feedback*, appears in MPC in an indirect way. Somewhere between steps 3 and 4, a measurement of the true system must take place. Unlike classical control—where this measurement process appears centrally with the notions of signals, closed-loop poles, and small gain theorems—optimization-based control treats stability through foreign concepts like *terminal costs*, *control-Lyapunov functions*, *invariant sets*, and *tubes*. Furthermore, the notion of feedback is made even more difficult to track by the inherent discrete time (or epoch) nature of the optimization-based control.

In other words, the frequency domain currency is no good here. This makes sense, because frequency domain tools primarily operate on linear systems, while an optimization-based control strategy is inherently nonlinear. Thus, there is a very clear disconnect: on the one hand, control designers want to use MPC because it is so simple and works so well. On the other hand, they are wary because when applied without care to convergence and stability issues, MPC can fail spectacularly—often at a great (physical) cost.

To help alleviate some of these concerns, this thesis argues that the right way to think about optimization-based control is through bounds on the Lyapunov, or energy, function (Chapter 2). Lyapunov bounds help answer questions about *robustness*, *adaptation*, and *learning* through algorithms, specifically convex optimization algorithms. In so doing, some systems concepts are brought back to the optimization-based setting. Specifically, we enlist the help of *adaptive* and *robust control* to design approximate adaptive- or robust-by-construction control methods, which must be implemented online as convex optimization problems (Chapters 3–4).

We must use algorithms to find tractable bounds, because for all but the simplest linear systems, exact Lyapunov functions are notoriously difficult to compute. Ways to provide useful upper and lower bounds have frequently come up as a theme in the past. Although hints at the idea can probably be traced back to Bellman [Bel57], the most illuminating reference is a two-part series on dissipation theory by Willems [Wil72a, Wil72b], in which he describes the notion of an energy storage function and a dissipation inequality. The storage function is bounded below by the available storage and above by the required supply. Since then, many researchers have used these ideas to develop bounds-based control and estimation policies.

An emphasis on questions that can be answered efficiently online as opposed to offline

pervades our approach. More importantly, these bounding techniques lead to efficiently computed approximate policies, depending on the type of bound employed. By using bounds instead of the true Lyapunov function, we give up some optimality for efficient synthesis or guarantees about safety. Of course, if useful bounds cannot be obtained or implemented, usually because the class of Lyapunov function candidates is not rich enough, we must remain agnostic about the applicability of these methods. However, in many cases results speak for themselves—as shown when Lyapunov bounding techniques are applied to formal methods and hybrid systems (Chapters 3–4).

Closely related to the Lyapunov thinking that led to the success of adaptive control are notions of identifiability that can be exported to other fields, such as communication networks and machine learning. Part of this thesis concentrates on novel notions of identifiability and persistence of excitation. Classical schemes in system identification and adaptive control often rely on persistence of excitation to guarantee parameter convergence, which may be difficult to achieve with a single agent and a single input. However, by adding communication to the mix, we show that it is possible to obtain parameter convergence even if no single agent employs a persistently exciting input (Chapters 5–6).

Summary of contributions. Chapter 2 grew out of a course that my advisor graciously let me develop and teach during Spring 2015. While the main ideas date back to Lyapunov [Lya92], and are (by now) fairly standard in nonlinear control, the notation is updated for the modern mathematical palate, often following the language of linear matrix inequalities [BEFB94, Boy08] and dissipation inequalities [Wil72a, Wil72b]. Chapter 3 reviews MPC, and presents recent directions for fast implementation, referencing ideas coming from various software implementations. These optimization tools are then combined with approximate dynamic programming to obtain Lyapunov bound based policies. The closest works are [BT96, Ran99, dFR03, WOB14], however the applications and examples are new. Chapter 4 is a novel application of bounds-based techniques to approximately solve hybrid and temporal logic constrained problems, based on the paper:

> [PFTM16] I. Papusha, J. Fu, U. Topcu, and R. M. Murray. Automata theory meets approximate dynamic programming: Optimal control with temporal logic constraints. In *IEEE Conference on Decision and Control (CDC), submitted*. 2016.

Chapter 5 switches gears to adaptation and learning by describing robust adaptive control in the language of optimization and Lyapunov bounds developed in the previous chapters. This chapter explores novel reformulations and makes connections between learning theory and adaptive control. The fundamental departure from *e.g.*, [LW13, IF06] is to view adaptive control not just as the result of a clever zeroing of dissipation terms in a Lyapunov argument, but rather as the implementation of a specific (continuous-time or discrete-time) algorithm for solving a constrained convex optimization problem online. This view is made more explicit in Chapter 6, which introduces a new concept of networked adaptive systems by specializing to a specific subspace parameter consensus constraint and implementing a gradient flow. In this networked setting, we derive an identifiability criterion and adaptation algorithm that trades off time and space, resulting in provable parameter convergence even without persistence of excitation. This last chapter is based on the paper:

> [PLM14] I. Papusha, E. Lavretsky, and R. M. Murray. Collaborative system identification via parameter consensus. In *American Control Conference (ACC)*, pp. 13–19. June 2014.

Not directly referenced in this thesis are the following additional publications:

> [HPB14] M. B. Horowitz, I. Papusha, and J. W. Burdick. Domain decomposition for stochastic optimal control. In *IEEE Conference on Decision and Control (CDC)*, pp. 1866–1873. 2014.
>
> [PM15] I. Papusha and R. M. Murray. Analysis of control systems on symmetric cones. In *IEEE Conference on Decision and Control (CDC)*, pp. 3971–3976. December 2015.
>
> [FPMM16] S. S. Farahani, I. Papusha, C. McGhan, and R. M. Murray. Constrained autonomous satellite docking via differential flatness and model predictive control. In *IEEE Conference on Decision and Control (CDC), submitted*. 2016.

Chapter 2

Lyapunov Theory

2.1 Conserved and dissipated quantities

In this section we set out notation by discussing autonomous dynamical systems of the form

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0, \tag{2.1}$$

where $x(t) \in \mathbf{R}^n$ is the system state at time t, and $f : \mathbf{R}^n \to \mathbf{R}^n$ is a function that corresponds to the (infinitesimal) direction of evolution of the state: given a state space location $x \in \mathbf{R}^n$, the quantity $f(x)$ is a tangent vector that points in the direction of the trajectory, see Figure 2.1. We assume that the initial value problem (2.1) has a unique solution $x(t)$ for all $t \geq 0$. This can be ensured, for example, if f is a globally, uniformly Lipschitz continuous function of its argument, see *e.g.*, [CL55, Per01].

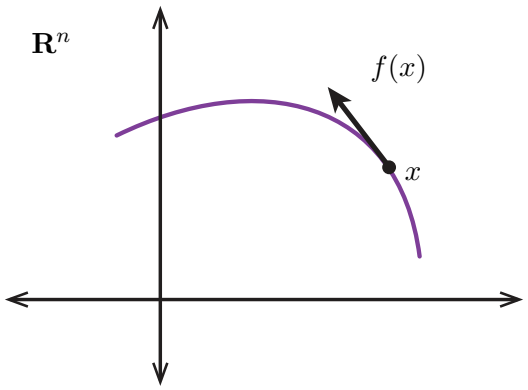


Figure 2.1: Schematic of general autonomous system $\dot{x} = f(x(t))$.

In control and system theory, the specific form of f may be too complicated, or perhaps even unknown. Thus the trajectories $x(t)$ may be too complicated to describe in closed

form. However, we would still like to quantitatively analyze the stability or dissipativity of the system (2.1) without reference to specific trajectories of f. An energy or *Lyapunov* function allows us to describe the behavior of the system (2.1) in a quite general way. For the remainder of this section, let $V : \mathbf{R}^n \to \mathbf{R}$ be a given real-valued function of state space.

Level sets. Given a real scalar $\alpha \in \mathbf{R}$, the *α-level set* L_α of V is the set

$$L_\alpha = \{z \in \mathbf{R}^n \mid V(z) = \alpha\}.$$

Conserved quantities. We say that V is a *conserved quantity* if it is constant along trajectories of (2.1). The quantity V is *conserved* if its time derivative does not change,

$$\frac{d}{dt}V(x(t)) = \nabla V(x(t))^T f(x(t)) = 0, \tag{2.2}$$

for all t. If V is a conserved quantity, then trajectories of (2.1) stay in level sets of V. To see why, suppose $V(x(0)) = \alpha$. Integrating $\dot{V}$ along the trajectories of (2.1) gives

$$\begin{aligned} V(x(t)) &= V(x(0)) + \int_0^t \dot{V}(x(\tau))\, d\tau \\ &= \alpha + \int_0^t \underbrace{\nabla V(x(\tau))^T f(x(\tau))}_{=0}\, d\tau \\ &= \alpha, \end{aligned}$$

for all $t \geq 0$. Thus if $x(0) \in L_\alpha$ and V satisfies the condition (2.2), then $x(t) \in L_\alpha$ for all $t \geq 0$.

Sublevel sets. Given a real scalar $\alpha \in \mathbf{R}$, the *α-sublevel set* S_α of V is the set

$$S_\alpha = \{z \in \mathbf{R}^n \mid V(z) \leq \alpha\}.$$

By definition, $L_\alpha = \mathbf{bd}\, S_\alpha$, hence the α-sublevel set S_α contains L_α, as well as all β-level and β-sublevel sets for which $\beta \leq \alpha$. Depending on the specific form of V, the sublevel sets S_α may be bounded or unbounded. If V is a convex function, then S_α is a convex set. See Figure 2.2.

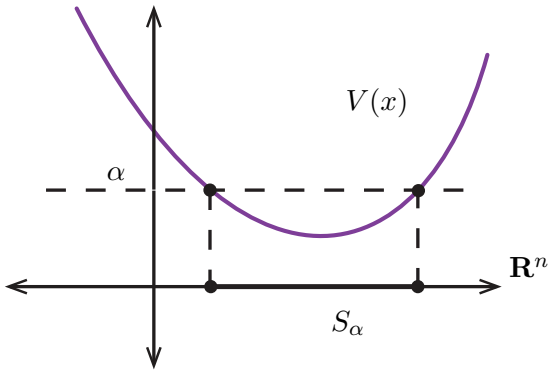


Figure 2.2: Sublevel sets S_α given an energy function $V(x)$.

Dissipated quantities. We say that V is a *dissipated quantity* if it is nonincreasing along trajectories of (2.1). The quantity V is *dissipated* if its time derivative is nonpositive,

$$\frac{d}{dt}V(x(t)) = \nabla V(x(t))^T f(x(t)) \leq 0, \tag{2.3}$$

and *strictly dissipated* if its time derivative is negative,

$$\frac{d}{dt}V(x(t)) = \nabla V(x(t))^T f(x(t)) < 0, \tag{2.4}$$

for all t. A key property of dissipated quantities is that trajectories of (2.1) stay in sublevel sets of V. To see why, suppose $V(x(0)) = \alpha$. Integrating $\dot{V}$ along the trajectories of (2.1) gives

$$\begin{aligned} V(x(t)) &= V(x(0)) + \int_0^t \dot{V}(x(\tau))\, d\tau \\ &= \alpha + \int_0^t \underbrace{\nabla V(x(\tau))^T f(x(\tau))}_{\leq 0}\, d\tau \\ &\leq \alpha, \end{aligned}$$

for all $t \geq 0$. Thus if $x(0) \in S_\alpha$ and V satisfies the condition (2.3), then $x(t) \in S_\alpha$ for all $t \geq 0$. The quantity $-\dot{V}$ is the *dissipation function*.

Example 2.1 *Spring-mass-dashpot.* The system illustrated in Fig 2.3 below has the linear dynamics

$$\underbrace{\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}}_{\dot{x}(t)} = \underbrace{\begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{c}{m}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}}_{f(x(t))},$$

where $x_1(t)$ is the signed displacement of the mass from its equilibrium position, $x_2(t) = \dot{x}_1(t)$ is its velocity, m is the mass, k is the spring constant, and c is dashpot constant.

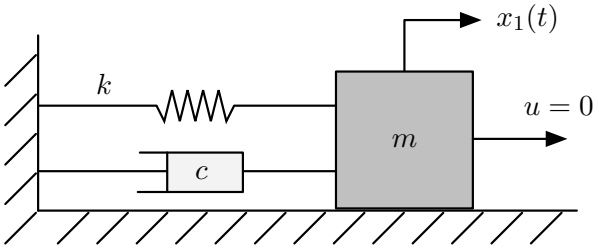


Figure 2.3: Spring-mass-dashpot system.

Define $V(x_1, x_2) = \frac{1}{2}kx_1^2 + \frac{1}{2}mx_2^2$ as the total energy (kinetic plus potential) of the mechanical system. The energy derivative is

$$\dot{V}(x_1, x_2) = \begin{bmatrix}kx_1\\ mx_2\end{bmatrix}^T \left(\begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{c}{m}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}\right) = -cx_2^2.$$

The total energy V is conserved if $c = 0$, and dissipated if $c > 0$. Note that by our definition, the total energy is not necessarily strictly dissipated, because $V(x_1, 0) = 0$ for all x_1.

2.2 Lyapunov's stability theorem

Lyapunov's insight. We should note that the conservation and dissipation conditions (2.2), (2.3), and (2.4) can be interpreted in two ways. First, we can think of $V(x(t))$ as a quantity that depends on a given trajectory $x(t)$. The time derivative $\dot{V}(x(t))$ is therefore computed indirectly through $x(t)$. This way of thinking, however, is not particularly useful, because it requires knowing the full trajectory $x(t)$.

Provided the appropriate smoothness conditions are met (*e.g.*, continuous differentiability of V and f as a function of x) and the chain rule applies, a second, more useful interpretation can be made: the quantity V is computed over the entire state space $\mathbf{R}^n$. Here, the conservation or dissipation of V depends only on the form of V and f, rather than on any specific trajectory. To this end, we often rewrite the conservation and dissipation

conditions (2.2), (2.3), and (2.4) as

$$\nabla V(x)^T f(x) = 0, \quad \forall x \in \mathbf{R}^n \tag{2.2'}$$

$$\nabla V(x)^T f(x) \leq 0, \quad \forall x \in \mathbf{R}^n \tag{2.3'}$$

$$\nabla V(x)^T f(x) < 0, \quad \forall x \in \mathbf{R}^n \setminus \{0\} \tag{2.4'}$$

to stress that V is a function on the state space $\mathbf{R}^n$, rather than of a specific trajectory $x(t)$. Thus whether or not V is conserved or dissipated can be concluded by checking the conditions (2.2′), (2.3′), or (2.4′).

The power of this small shift in thinking, widely attributed to Lyapunov's 1892 thesis [Lya92] has not gone unnoticed in the dynamical systems community. As we will see, the power of this shift lies in an ability to quantify stability, optimality, and robustness of control policies. Lyapunov is arguably the principal enabler of modern control.

Generalized energy. A function $V : \mathbf{R}^n \to \mathbf{R}$ is *positive definite* if it satisfies the following conditions:

- $V(x) \geq 0$ for all $x \in \mathbf{R}^n$,
- $V(x) = 0$ if and only if $x = 0$,
- All sublevel sets of V are bounded.

A positive definite function is a kind of *generalized energy* for the system. As Lyapunov's famous result asserts, the existence of a strictly dissipated generalized energy is a sufficient condition for the stability of the system.

> **Example 2.2** The quadratic function $V(x) = x^T P x$ is positive definite if and only if the matrix P is a positive definite matrix, $P \succ 0$.

Theorem 1 (Lyapunov, 1892)**.** *Suppose there is a function $V : \mathbf{R}^n \to \mathbf{R}$ such that*

- *V is positive definite (generalized energy)*
- *$\nabla V(x)^T f(x) < 0$ for all $x \neq 0$ and $\nabla V(0)^T f(0) = 0$ (strict dissipation)*

then every trajectory of $\dot{x}(t) = f(x(t))$ converges to zero as $t \to \infty$.

Proof. Suppose $x(t) \not\to 0$ as $t \to \infty$. Since V is a dissipated, nonnegative quantity, the conditions $V(x(t)) \geq 0$ and $\dot{V}(x(t)) \leq 0$ together mean that $V(x(t))$ is a monotone, bounded function of t, therefore $V(x(t)) \to c_1 > 0$ as $t \to \infty$ for some positive constant c_1. In particular, $c_1 \leq V(x(t)) \leq V(x(0)) = c_2$ for all $t \geq 0$. Take

$$C = \{z \in \mathbf{R}^n \mid c_1 \leq V(z) \leq c_2\}.$$

Since C is a compact subset of S_{c_2}, which excludes a neighborhood of the origin, and V is strictly dissipated, we have $\sup_{z \in C} \dot{V}(z) = -\gamma < 0$ for some positive constant γ. But the energy at time t is given by

$$V(x(t)) = V(x(0)) + \int_0^t \underbrace{\dot{V}(x(\tau))}_{\leq -\gamma} \, d\tau \leq c_2 - \gamma t,$$

which is negative for large t, leading to a contradiction. □

Example 2.3 Consider the linear system $\dot{x} = Ax$, where $A \in \mathbf{R}^{n \times n}$, and define $V(x) = x^T P x$, where $P = P^T \in \mathbf{R}^{n \times n}$ is a given symmetric matrix. The derivative along trajectories of $\dot{x} = Ax$ is given by

$$\begin{aligned} \dot{V}(x) &= \dot{x}^T P x + x^T P \dot{x} \\ &= (Ax)^T P x + x^T P (Ax) \\ &= x^T (A^T P + P A) x. \end{aligned}$$

Thus, we conclude the following:

- V is positive definite if and only if $P \succ 0$.
- V is strictly dissipated if and only if $A^T P + P A \prec 0$.

We can restate Lyapunov's theorem in the linear setting as follows: if there exists a matrix $P \succ 0$ with $A^T P + P A \prec 0$, then all trajectories of $\dot{x} = Ax$ converge to the origin as $t \to \infty$.

Necessary condition. Note that Theorem 1 is a sufficiency result. It states that if a positive definite Lyapunov function exists, then the system can be certified as stable. It states nothing about how to compute the Lyapunov function. A number of so-called *converse* theorems have been proposed [KUR55A, KUR55B, MAS56], which state the technical conditions under which a Lyapunov function must exist to prove the stability of a dynamical

system; see, *e.g.*, [KRA63, LSW96]. One example is linear systems, for which it is easy to see that a quadratic Lyapunov function is sufficient to prove stability.

Example 2.4 *Converse Lyapunov result for linear systems.* Let $Q = Q^T \succ 0$ be any positive definite matrix and let $\dot{x} = Ax$ be a (Hurwitz) stable system. Let

$$P = \int_0^\infty e^{A^T\tau} Q e^{A\tau}\, d\tau,$$

which is well defined because all eigenvalues of A have negative real part. Consider the quadratic Lyapunov function $V(x) = x^T P x$.

$$\begin{aligned}
\dot{V}(x(t)) &= x(t)^T (A^T P + PA) x(t) \\
&= x(t)^T \left(A^T \int_0^\infty e^{A^T\tau} Q e^{A\tau}\, d\tau + \int_0^\infty e^{A^T\tau} Q e^{A\tau}\, d\tau A \right) x(t) \\
&= x(t)^T \left(\int_0^\infty \frac{d}{dt} \left\{ e^{A^T\tau} Q e^{A\tau} \right\} d\tau \right) x(t) \\
&= -x(t)^T Q x(t)
\end{aligned}$$

for any trajectory $x(t)$ of the system. Thus the quantity V is positive definite and strictly dissipated. The matrix P, also known as a *Gramian*, satisfies the Lyapunov equation $A^T P + PA + Q = 0$.

Example 2.5 *Certificate of instability.* Let $\dot{x} = Ax$ be an autonomous linear system and suppose there exists a function $V(x) = x^T P x$ with $P \not\succeq 0$ such that $A^T P + PA \preceq 0$. Then A is not (Hurwitz) stable.

To see this, note that the condition $A^T P + PA \preceq 0$ implies that all trajectories of the autonomous system $\dot{x} = Ax$ satisfy $V(x(t)) \leq V(x(0))$ for all $t \geq 0$, because

$$\begin{aligned}
V(x(t)) - V(x(0)) &= \int_0^t \dot{V}(x(\tau))\, d\tau \\
&= \int_0^t \underbrace{x(\tau)^T (A^T P + PA) x(\tau)}_{\leq 0}\, d\tau \leq 0.
\end{aligned}$$

But since $P \not\succeq 0$, there exists a vector w such that $V(w) < 0$. Setting the initial condition $x(0) = w$, we have

$$V(x(t)) \leq V(w) < 0, \quad \text{for all} \quad t \geq 0.$$

In particular, $\liminf_{t\to\infty} \|x(t)\|_2 \neq 0$.

Graphical interpretation. We can interpret Lyapunov's theorem graphically by plotting the level and sublevel sets of the function V. A schematic view is shown in Figure 2.4. Here, the solid line denotes a particular level set $L_\alpha = \{z \in \mathbf{R}^n \mid V(z) = \alpha\}$ for a given scalar α, and the shaded area inside is the sublevel set $S_\alpha = \{z \in \mathbf{R}^n \mid V(z) \leq \alpha\}$. If V is positive definite, then S_α is bounded. The arrowed line corresponds to a particular trajectory $x(t)$.

For a given state x, the inner product between the vectors $\nabla V(x)$ and $f(x)$ determines whether a trajectory through x will enter the appropriate level or sublevel set. For example, if the inner product is zero ($\nabla V(x)^T f(x) = 0$, V is conserved), then a trajectory through x will follow the level set of V corresponding to $V(x) = \alpha$. If the inner product is nonpositive, ($\nabla V(x)^T f(x) \leq 0$, V is dissipated), then the trajectory through x cannot escape the sublevel set S_α. Finally, if the inner product is strictly negative ($\nabla V(x)^T f(x) < 0$, V is strictly dissipated), then the trajectory through x must enter $\mathbf{int}\, S_\alpha$.

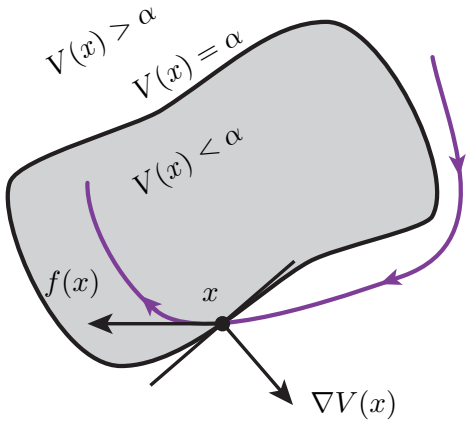


Figure 2.4: Graphical interpretation of Lyapunov's theorem.

As the following example illustrates, some care must be taken to ensure strict dissipation when considering the stability of a system.

Example 2.6 *Non-strict dissipation.* If it can only be concluded that $\dot{V}(x) \leq 0$ but not $\dot{V}(x) < 0$, then trajectories can "hide" in the zero-dissipation set

$$\mathcal{Z} = \{z \in \mathbf{R}^n \mid \dot{V}(z) = 0\}.$$

Let $\dot{x} = Ax$ be an autonomous linear system and consider $V(x) = x^T P x$ with the specific constants

$$A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

In this case, V is positive definite, and dissipated (but not strictly dissipated), because $A^T P + PA = 0$. The zero-dissipation set is $\mathcal{Z} = \mathbf{R}^2$. In fact, this "center" system has circular trajectories that do not approach the origin, unless they already start there.

Decay rate. If $V(x) = x^T P x$ is positive definite and dissipated with the further restriction that $\dot{V}(x) \leq -2\alpha V(x)$, then trajectories of (2.1) decay exponentially with rate at least α,

$$\lim_{t\to\infty} e^{\alpha t}\|x(t)\|_2 = 0.$$

In this case, the scalar α is a *Lyapunov exponent* for the system. To see why the decay rate is at least α, recall Grönwall's inequality, which states

$$\dot{h}(t) \leq g(t)h(t) \text{ for all } t \in (a,b) \quad \Longrightarrow \quad h(t) \leq h(a)\exp\left\{\int_a^t g(\tau)\,d\tau\right\} \text{ for all } t \in (a,b).$$

Applying Grönwall's inequality to $V(x(t))$ gives the bound

$$V(x(t)) \leq V(x(0))\exp\left\{-\int_0^t 2\alpha\,d\tau\right\},$$

which means $x(t)^T P x(t) \leq x(0)^T P x(0) e^{-2\alpha t}$ for all $t \geq 0$. Therefore,

$$\|x(t)\|_2 \leq \sqrt{\frac{\sigma_{\max}(P)}{\sigma_{\min}(P)}}\|x(0)\|_2 e^{-\alpha t}, \quad \text{for all } t \geq 0.$$

Evidently, such a Lyapunov function certifies that the decay rate is at least α.

Region of attraction. Lyapunov functions can also be used to estimate regions of attraction by comparing the regions of state space over which the Lyapunov function is strictly dissipated against the Lyapunov function's sublevel sets. Define the region of attraction

$$\mathcal{R} = \{x_0 \in \mathbf{R}^n \mid \lim_{t\to\infty} x(t) = 0\}$$

as the set of initial conditions in (2.1) for which the trajectory through that initial condition approaches the origin. Next, define the strict dissipation set

$$\mathcal{D} = \{z \in \mathbf{R}^n \mid \dot{V}(z) < 0\} \cup \{0\}.$$

Trajectories starting at a point $x_0 \in \mathcal{D}$ with initial energy $V(x_0) = \alpha$ must initially stay within S_α, because V is dissipated within $\mathcal{D}$. If S_α happens to contain a point outside $\mathcal{D}$, then a trajectory through that point can gain energy and escape S_α, because dissipation of V is only guaranteed within $\mathcal{D}$. However, if S_α is entirely within $\mathcal{D}$, then no trajectory can escape S_α. Therefore, $S_\alpha \subseteq \mathcal{R}$ is an inner approximation of the region of attraction $\mathcal{R}$ if it can be shown that $S_\alpha \subseteq \mathcal{D}$. Depending on the problem, it may be easier to show that $S_\alpha \subseteq \mathcal{D}$ (for example, by semidefinite programming) than to compute $\mathcal{R}$ explicitly. See Example 2.7 and Figure 2.5 for an illustration.

Example 2.7 *Van der Pol oscillator.* Consider the dynamics

$$\begin{cases} \dot{x}_1 = -x_2 \\ \dot{x}_2 = x_1 + (x_1^2 - 1)x_2, \end{cases}$$

which are locally stable about the equilibrium $(0, 0)$. The region of attraction $\mathcal{R}$ corresponding to this equilibrium is enclosed by the limit cycle (Figure 2.5). Using a quadratic Lyapunov function of the form $V(x) = x^T P x$,

$$V(x) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \underbrace{\begin{bmatrix} 1.5 & -0.5 \\ -0.5 & 1 \end{bmatrix}}_{P} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad A^T P + PA = -I \prec 0,$$

we can form an ellipsoidal estimate of $\mathcal{R}$ as follows. First, we determine the strict dissipation region $\mathcal{D}$. Then, we search for the maximum scalar α for which the sublevel set S_α is entirely contained in $\mathcal{D}$. In this case, the largest such sublevel set corresponds to $S_{2.25}$. Therefore, $S_{2.25}$ is an inner approximation of $\mathcal{R}$.

Lyapunov stability for linear systems. Linear systems and quadratic Lyapunov functions have a special relationship. Generically speaking, a Lyapunov function can always be constructed for any stable system assuming f is Lipschitz [KUR55A, KUR55B], but its form may be too complicated to describe exactly. A special property of linear systems is that a quadratic Lyapunov function is "enough." For example, if a linear system is stable, then there exists a quadratic Lyapunov function to prove it. If a system is unstable, then there exists a quadratic function to prove that, as well. See Examples 2.4 and 2.5.

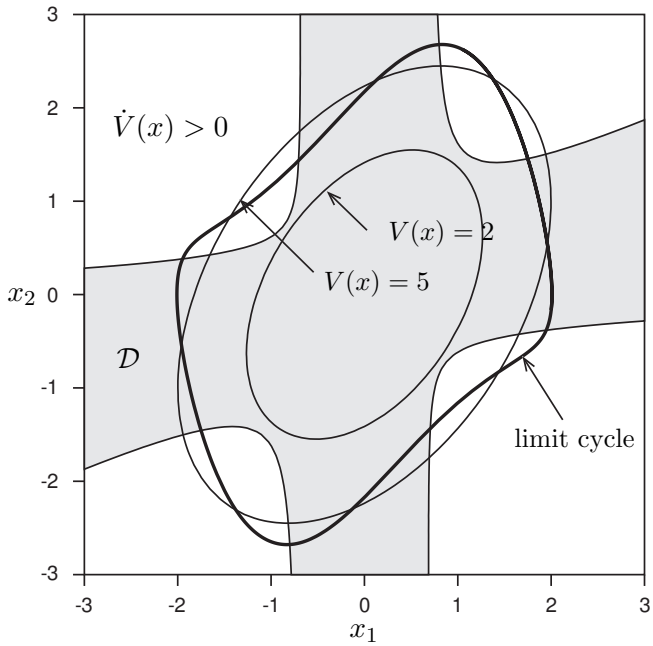


Figure 2.5: Van der Pol region of attraction estimate. The strict dissipation set (shaded) is $\mathcal{D} = \{z \mid \dot{V}(z) < 0\} \cup \{0\}$, with the largest ellipsoidal sublevel set contained in $\mathcal{D}$ given by $S_\alpha = \{z \mid V(z) \leq 2.25\}$. The true region of attraction $\mathcal{R}$ is the area enclosed by the limit cycle.

Theorem 2. *For the state space system $\dot{x} = Ax$, $V(x) = x^T P x$, and*

$$\dot{V}(x) = x^T(A^T P + PA)x = -x^T Q x,$$

if $P \succ 0$ and $Q \succ 0$, then $x(t) \to 0$. Conversely, if $\dot{x} = Ax$ is (Hurwitz) stable, then there exists $P \succ 0$ and $Q \succ 0$ to prove it.

Results for linear systems are also typically global. Consider a linear system $\dot{x} = Ax$. For a quadratic Lyapunov function $V(x) = x^T P x$, the energy sublevels are (possibly degenerate) ellipsoids,

$$S_\alpha = \{z \in \mathbf{R}^n \mid z^T P z \leq \alpha\}.$$

If the system is stable, the dissipation sets are all of $\mathbf{R}^n$,

$$\mathcal{D} = \{z \in \mathbf{R}^n \mid \dot{V}(z) = z^T(A^T P + PA)z \leq 0\}.$$

Since $S_\alpha \subseteq \mathcal{D} = \mathbf{R}^n$ for all α, we conclude that state space systems are either globally stable, or not globally stable.

This connection between quadratic Lyapunov functions and linear systems allows the

study of linear systems to be viewed as the study of invariant ellipsoids and the sublevels of an appropriate quadratic function, or more usefully, of the associated quadratic form. Many textbooks have been devoted to this subject [FK96, Son98, Kha02], and it is not our intention to reproduce that work here. The interested reader is referred to the textbook [Kha02] for a general treatment of linear and nonlinear stability theory, and to [DP00] for a more control theoretic spin. This thesis builds on these works, but specifically uses the conventions and notation of [BEFB94, FK96].

2.3 Robust control

2.3.1 Linear differential inclusions

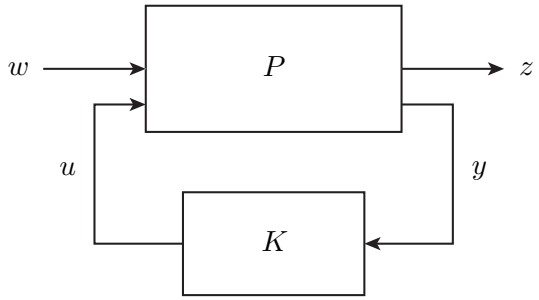


Figure 2.6: Robust controller synthesis.

A linear system with unknown or uncertain parameters can be described as a state space model

$$\begin{aligned} \dot{x} &= A(t)x + B_u(t)u + B_w(t)w \\ z &= C_z(t)x + D_{zu}(t)u + D_{zw}(t)w \\ x(0) &= x_0, \end{aligned} \tag{2.5}$$

where $x \in \mathbf{R}^{n_x}$ is the state, $u \in \mathbf{R}^{n_u}$ is the control input, $w \in \mathbf{R}^{n_w}$ is an exogenous input, and $z \in \mathbf{R}^{n_z}$ is the regulated output. The goal of a typical robust control problem is to choose a control input u as a function of a measured output y that minimizes some cost index that relates the inputs, outputs, and disturbances. For simplicity we will consider *state-feedback* systems, in which case the measured output is $y = x$. Key to the approach

is the assumption that the model matrices obey

$$\begin{bmatrix} A(t) & B_u(t) & B_w(t) \\ C_z(t) & D_{zu}(t) & D_{zw}(t) \end{bmatrix} \in \Omega, \quad \text{for all } t,$$

for a given description of the set $\Omega \subseteq \mathbf{R}^{(n_x+n_z)\times(n_x+n_u+n_w)}$. Such a description is called a Linear Differential inclusion (LDI).

LTI systems. The set Ω might consist of a single element,

$$\Omega = \left\{ \begin{bmatrix} A & B_u & B_w \\ C_z & D_{zu} & D_{zw} \end{bmatrix} \right\},$$

in which case the system is known fully. It is still subject to disturbances w, however the design procedure for robust controllers for LTI systems is considerably simplified, at most requiring the solution of linear (Lyapunov) or quadratic (Riccati) matrix equations.

Polytopic systems. In this case, the set Ω might be described by the vertices of a polygon, *i.e.*,

$$\Omega = \mathbf{Co} \left\{ \begin{bmatrix} A_1 & B_{u,1} & B_{w,1} \\ C_{z,1} & D_{zu,1} & D_{zw,1} \end{bmatrix}, \ldots, \begin{bmatrix} A_L & B_{u,L} & B_{w,L} \\ C_{z,L} & D_{zu,L} & D_{zw,L} \end{bmatrix} \right\}.$$

Optimal controller synthesis with a common Lyapunov function for polytopic model uncertainties usually requires semidefinite programming.

Norm-bound systems. A very useful uncertainty class consists of model descriptions of the form

$$\Omega = \left\{ \tilde{A} + \tilde{B}\Delta(I - D_{qp}\Delta)^{-1}\tilde{C} \mid \|\Delta\|_2 \leq 1 \right\}, \tag{2.6}$$

where

$$\tilde{A} = \begin{bmatrix} A & B_u & B_w \\ C_z & D_{zu} & D_{zw} \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_p \\ D_{zp} \end{bmatrix}, \quad \tilde{C} = \begin{bmatrix} C_q & D_{qu} & D_{qw} \end{bmatrix}.$$

The set Ω is the image of the matrix unit ball under the matrix linear-fractional mapping

$$\Delta \mapsto \tilde{A} + \tilde{B}\Delta(I - D_{qp}\Delta)^{-1}\tilde{C}, \tag{2.7}$$

and is *well-posed* provided $D_{qp}^T D_{qp} \prec I$. The norm-bound model uncertainty corresponds to a linear state-feedback perturbation $\Delta(t)$ being applied to an LTI system as shown in Figure 2.7.

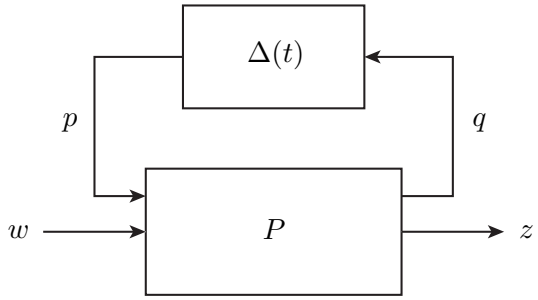


Figure 2.7: Norm-bound perturbation.

The specific way this perturbation is applied to (2.5) is interpreted as follows. We can rewrite the uncertain dynamics (2.5) subject to a norm-bound feedback perturbation as the dynamical system,

$$\begin{aligned} \dot{x} &= Ax + B_p p + B_u u + B_w w \\ q &= C_q x + D_{qp} p + D_{qu} u + D_{qw} w \\ z &= C_z x + D_{zp} p + D_{zu} u + D_{zw} w \\ p &= \Delta(t) q, \quad \|\Delta(t)\|_2 \leq 1 \\ x(0) &= x_0. \end{aligned} \tag{2.8}$$

In this state-space description (2.8), a signal q is "picked off" from the original LTI dynamics (2.5). That signal is then fed through a gain-bounded block to generate an uncertain signal $p = \Delta(t)q$. The signal p is then applied back to the dynamics (2.5). To see that (2.7) is the correct mapping for the norm-bound perturbation, we rewrite (2.8) in matrix form,

$$\begin{bmatrix} \dot{x} \\ z \\ q \end{bmatrix} = \begin{bmatrix} A & B_u & B_w & B_p \\ C_z & D_{zu} & D_{zw} & D_{zp} \\ C_q & D_{qu} & D_{qw} & D_{qp} \end{bmatrix} \begin{bmatrix} x \\ u \\ w \\ p \end{bmatrix}, \quad p = \Delta(t)q, \quad \|\Delta(t)\|_2 \leq 1. \tag{2.9}$$

The last row of (2.9) can be eliminated. With $p = \Delta(t)q$, the last row reads

$$\begin{aligned} q &= \begin{bmatrix} C_q & D_{qu} & D_{qw} \end{bmatrix} \begin{bmatrix} x \\ u \\ w \end{bmatrix} + D_{qp}\Delta(t)q \\ &= \tilde{C} \begin{bmatrix} x \\ u \\ w \end{bmatrix} + D_{qp}\Delta(t)q, \end{aligned} \tag{2.10}$$

so if $(I - D_{qp}\Delta(t))$ is invertible for all t, we can determine q from (2.10) by taking the inverse,

$$q = (I - D_{qp}\Delta(t))^{-1}\tilde{C} \begin{bmatrix} x \\ u \\ w \end{bmatrix}. \tag{2.11}$$

Substituting (2.11) back into (2.9) gives

$$\begin{bmatrix} \dot{x} \\ z \end{bmatrix} = \left(\tilde{A} + \tilde{B}\Delta(t)(I - D_{qp}\Delta(t))^{-1}\tilde{C} \right) \begin{bmatrix} x \\ u \\ w \end{bmatrix},$$

which gives the uncertainty description (2.6).

The well-posedeness assumption

$$I - D_{qp}\Delta(t) \succ 0, \quad \|\Delta(t)\|_2 \leq 1 \quad \text{for all } t,$$

restricts the domain of the mapping (2.7) so the set Ω is convex. By making use of the norm bound $\|\Delta(t)\|_2 \leq 1$, equivalently the map (2.7) is well posed if and only if $D_{qp}^T D_{qp} \prec I$. This assumption is often satisfied by removing the dependence of q on p by setting $D_{qp} = 0$.

2.3.2 Quadratic stability margins.

Lyapunov theory and LDI descriptions of model uncertainties, particularly norm-bound LDIs, are very useful for computing quadratic stability margins. A quadratic stability margin gives a sense of how much the model uncertainty set Ω can be expanded while still

remaining quadratically stable.

Guaranteed LQR margins. Consider the state feedback system

$$\begin{aligned} \dot{x} &= Ax + B\Delta u \\ u &= Kx, \end{aligned} \tag{2.12}$$

where $K = -R^{-1}B^TP$ is an LQR gain, and $P \succ 0$ satisfies the algebraic Riccati equation

$$A^TP + PA - PBR^{-1}B^TP + Q = 0$$

for given matrices $Q \succ 0$, $R \succ 0$. Assume that R is diagonal and the matrix $\Delta \in \mathbf{R}^{n_u \times n_u}$ is a diagonal multiplicative perturbation to the input of a nominal system. The system (2.12) is quadratically stable with Lyapunov function $V(x) = x^TPx$ if the matrix Δ satisfies $\Delta_{ii} \geq 1/2$ for all $i = 1, \ldots, n_u$.

To see why, we can use the Riccati equation to show that

$$\dot{V}(x) = x^T(PB(R^{-1} - R^{-1}\Delta^T - \Delta R^{-1})B^TP - Q)x < 0 \quad \text{for all } x \neq 0$$

provided $R^{-1} - R^{-1}\Delta^T - \Delta R^{-1} \preceq 0$, which happens if Δ is diagonal with $\Delta_{ii} \geq 1/2$. This corresponds to a classical lower gain margin of

$$20\log_{10}(1/2) = -6\text{dB}$$

and an upper gain margin of ∞. Thus, LQR is robust to $1/2$ gain reduction and unbounded gain amplification in each input channel.

LQR margin shaping. We can view the LQR state feedback $u = (I + \Delta)Kx$ with an uncertain additive feedback perturbation by rewriting (2.12) as

$$\begin{aligned} \dot{x} &= Ax + B_u u + B_p p \\ q &= C_q x \\ p &= \Delta q, \quad \|\Delta\|_2 \leq \alpha, \end{aligned} \tag{2.13}$$

where $B_p = B_u$ and $C_q = K$ are fixed, and the nominal control input is $u = Kx$. If the input uncertainty is norm-bounded by α (and not necessarily diagonal), with $\|\Delta\|_2 \leq \alpha$, we can ask the question of for which α is (2.13) quadratically stable. Equivalently, with the Lyapunov function $V(x) = x^T P x$, we ask when is the quadratic form

$$\dot{V}(x) = \begin{bmatrix} x \\ p \end{bmatrix} \begin{bmatrix} (A + B_u K)^T P + P(A + B_u K) & PB_p \\ B_p^T P & 0 \end{bmatrix} \begin{bmatrix} x \\ p \end{bmatrix} \leq 0,$$

for all (x, p) satisfying (2.13). From the S-procedure, we conclude that (2.13) provided the matrix inequality

$$P \succ 0, \quad \tau \geq 0, \quad \begin{bmatrix} (A + B_u K)^T P + P(A + B_u K) + \tau \alpha^2 C_q^T C_q & PB_p \\ B_p^T P & -\tau I \end{bmatrix} \preceq 0$$

is feasible. Substituting the Riccati equation and $K = -R^{-1} B^T P$, this is equivalent to

$$P \succ 0, \quad \tau \geq 0, \quad \begin{bmatrix} -Q - PB_u R^{-1}(I - \tau \alpha^2 R^{-1}) B_u^T P & PB_u \\ B_u^T P & -\tau I \end{bmatrix} \preceq 0,$$

which is feasible provided $\alpha^2 \leq \lambda_{\min}(R)/\lambda_{\max}(Q^{-1/2} P B^T P Q^{-1/2})$.

2.4 Fundamental variational bounding ideas

One of the most powerful applications of Lyapunov theory to control lies in two very simple bounds. One of these bounds gives an upper bound to the Lyapunov function, and the other gives a lower bound. Although they differ only in the sense of inequality, and are obtained in effectively the same way, these two bounds lead to very different interpretations of the Lyapunov function.

To see how this kind of bounding works, we consider the autonomous system

$$\dot{x}(t) = A(t)x(t), \quad x(0) = x_0, \tag{2.14}$$

where at each time instant t, the matrix $A(t)$ is chosen from a convex set $\Omega \subset \mathbf{R}^{n \times n}$, and the initial condition $x_0 \in \mathbf{R}^n$ is given. For simplicity, we assume that solutions to (2.14) are unique given the function $A(\cdot)$.

One might think of the choice of $A(t)$ as being made nondeterministically by an adversarial *nature*, in which case the specific value of $A(t)$ is unknown beyond the uniqueness and set membership constraints, or by the *designer*, in which case the value $A(t)$ is perhaps determined by a control algorithm. Let

$$J = \int_0^\infty \ell(x(t))\, dt$$

denote the cost of a given trajectory $x(\cdot)$ satisfying (2.14), where $\ell(x) \geq 0$ for all $x \in \mathbf{R}^n$. Note that J might be infinite if the system (2.14) is unstable.

Now suppose there exists a continuously differentiable Lyapunov function $V^+ : \mathbf{R}^n \to \mathbf{R}$ that satisfies

$$V^+(x) \geq 0, \quad \dot{V}^+(x) \leq -\ell(x), \tag{2.15}$$

for all $x \in \mathbf{R}^n$ and for all $A(t) \in \Omega$. Then for a fixed x_0, the quantity $V^+(x_0)$ is an *upper bound* on J. To see this, we integrate along the trajectory

$$\begin{aligned} V^+(x(T)) - V^+(x(0)) &= \int_0^T \underbrace{\dot{V}^+(x(t))}_{\leq -\ell(x(t))}\, dt \\ &\leq -\int_0^T \underbrace{\ell(x(t))}_{\geq 0}\, dt \\ &\leq -\int_0^\infty \ell(x(t))\, dt = -J, \end{aligned}$$

where the first inequality follows from (2.15), and the second from the nonnegativity of the loss function ℓ. Therefore, we have

$$J \leq V^+(x(0)) - V^+(x(T)) \leq V^+(x_0), \qquad \text{for all } T,$$

because $V^+(x(T))$ is nonnegative for all T. Thus $V^+(x_0)$ is an upper bound on the worst-case cost of the trajectory for any valid choice of the values $A(t)$ that nature might make.

In fact, we have the variational inequality

$$J \leq J^{\text{wc}} = \inf_{V^+ \text{ satisfies (2.15)}} V^+(x_0), \tag{2.16}$$

therefore no choice that nature might make could ever make the cost bigger than the worst

case value $J^{\rm wc}$, a quantity obtained by minimizing $V^+(x_0)$ over all functions V^+ that satisfy the condition (2.15). Thus if we are able to compute the infimum in (2.16), and it is finite, then we have a quantifiable measure of the robustness of the cost with respect to the choice of $A(t)$ at every time instant t. It is sometimes possible to quantify this robustness given the description of the set Ω alone. For example, if Ω is consists of a single Hurwitz matrix element, then (2.14) must be stable, and J is no more than the initial energy. However, if nature wanted to be devious, it might pick $A(t) \in \operatorname{argmax}_{A\in\Omega} \dot{V}^+(x(t))$ as a function of $x(t)$ to make the cost as close to $J^{\rm wc}$ as possible.

On the other hand if the Lyapunov function satisfies the reverse inequality

$$V^-(x) \geq 0, \quad \dot{V}^-(x) \geq -\ell(x), \tag{2.17}$$

for all $x \in \mathbf{R}^n$ and for all $A(t) \in \Omega$, then the same chain of inequalities leads to

$$J \geq V^-(x_0) - V^-(x(T)),$$

for all T. Now suppose that the state trajectory $x(\cdot)$ is asymptotically stable, and $V^-(0) = 0$, then as $T \to \infty$ we have $V^-(x(T)) \to 0$. In other words, $V^-(x_0)$ is a *lower*, or performance bound for any stabilizing choice of the values $A(t)$. Similarly, we have the variational bound

$$J \geq J^{\rm perf} = \sup_{V^- \text{ satisfies (2.17)}} V^-(x_0), \tag{2.18}$$

therefore no stabilizing $A(t)$ can ever achieve the performance limitation $J^{\rm perf}$. To get a sequence that gives a small cost, a control designer might choose $A(t) \in \operatorname{argmin}_{A\in\Omega} \dot{V}^-(x(t))$ as a function of $x(t)$ to make the cost as close to the performance limit $J^{\rm perf}$ as possible.

Indeed, asking for an upper bound on the cost leads to a discussion of robustness and stability, and for a lower bound—a limit on performance. Thus the former is the basis of robust control, while the latter leads to suboptimal policies. It is a miracle of linear systems that these bounds can be implemented (often, in a tight way) as semidefinite programs or even Lyapunov and Riccati equations. However, even when they are not tight, these bounding ideas are extremely useful, and can be used to solve extremely complicated problems. Chapter 2 gives applications of the upper bound, and Chapters 3–4 give applications of the lower bound.

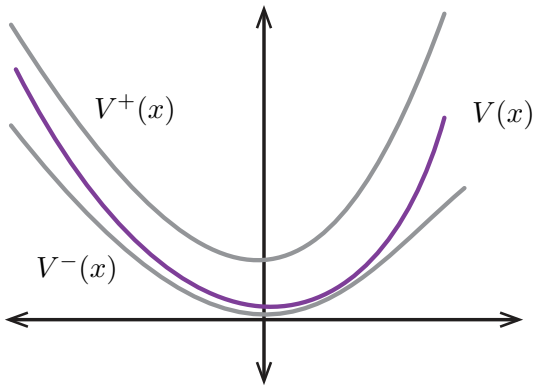


Figure 2.8: Upper and lower bounds on a Lyapunov function.

2.5 References

The idea of using a real-valued positive definite energy function to prove stability was described by Lyapunov more than a century ago [Lya92] (translation [Lya92]), and by now the theory is quite mature. Theorem 1, also known as Lyapunov's direct method, is a sufficiency result: it states that if a generalized energy function can be found, then a conclusion can be made about local or global stability of the equilibrium. To prove stability, one usually searches over a class (such as quadratic or sum-of-squares [Par00]) of generalized energy functions that satisfy dissipativity conditions. However, if a function that satisfies the dissipativity conditions is not found, the system is not necessarily unstable, except in the case of linear systems—for linear systems, it always suffices to search over quadratic functions. More generally, converse Lyapunov theorems, which appeared in [Kur55a, Kur55b, Mas56], gave conditions under which it is possible to construct Lyapunov functions. The Kalman–Yakubovich–Popov (KYP) lemma [Yak62] was one of the unifying results that linked quadratic Lyapunov functions with system theoretic concepts, such as $\mathbf{H}_\infty$-norm, and allowed searching for quadratic Lyapunov functions by semidefinite programming. A great modern proof of the KYP lemma was given by Rantzer in [Ran96]. A beautiful description of the concepts of supply rates and storage functions, including lower and upper bounds is given in a classic paper by Willems [Wil72a], with applications to linear quadratic theory in [Wil72b]. Searching for Lyapunov functions by convex optimization in general is often attributed to [PS82], and the concept of a Control Lyapunov Function [Son98]. The links between semidefinite programming, robust control, linear systems, and quadratic Lyapunov functions are reviewed in [BEFB94].

Chapter 3

Model Predictive and Approximately Optimal Control

3.1 Online optimization for control

In the previous chapter, we introduced the Lyapunov function as a fundamental tool for analyzing autonomous systems. Searching for a Lyapunov function that obeys the appropriate dissipation conditions results in a functional optimization problem, which for linear systems and quadratic Lyapunov function candidates is a finite dimensional semidefinite program over the coefficients of the Lyapunov function parameterization. For controlled systems, finding a Lyapunov function in general is difficult, but not impossible (*e.g.*, [Son98, Par00]), and the standard prescription involves the following computations:

1. *Offline*: find a Lyapunov function that is provably dissipated known as a Control Lyapunov Function (CLF)

2. *Online*: use the CLF to define a control policy that runs in the loop

We divide the two computations into online and offline parts to stress that the procedure to complete the first part is (in principle) allowed to take as much or as little time as needed, because a Lyapunov function needs only to be found once for a given set of dynamics. The only limiting factors are the tractability of the dynamics, and the designer's time, patience, or (sometimes) ingenuity.

For example, a typical approach for a controlled linear system would be to use a quadratic Lyapunov function candidate to determine a linear state feedback control law with specified quadratic stability margins. Once the linear feedback coefficients are found

after solving an SDP, the control law is fixed and can be computed online in the loop by appropriately mixing the state values. For nonlinear systems, a more complicated (*e.g.*, quartic, hexic, *etc.*) Lyapunov function candidate may be used, from which the polynomial control policy can be obtained by taking the appropriate gradients.

This offline-online control design procedure can fail at the first step if the parameterization of Lyapunov function candidates is not rich enough to encompass the desired closed loop dynamics, even if the closed loop dynamics can be achieved. For example, adding state and input constraints to linear systems is an easy way to ensure that linear state feedback and quadratic Lyapunov function candidates do not suffice.

An alternative is to conflate the offline-online procedure above into one online algorithm:

1. *Online*: calculate an optimal or approximately optimal control input in the loop by solving an optimization problem.

This is the idea used by Model Predictive Control (MPC), in which (for discrete time systems), the control input is computed at each time step by solving an optimization problem. The key requirements are that that optimization problem be solvable quickly enough (*i.e.*, the problem is convex and not too large-dimensional), and that the system dynamics are robust enough to deal with any suboptimalities that MPC may introduce. There is no need to search over a specific class of Lyapunov function candidates, because the optimization problem itself serves as a Lyapunov function.

A typical linear-quadratic MPC strategy would solve the following optimization at every timestep,

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\sum_{\tau=0}^{T-1}\left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau\right) + \frac{1}{2}x_T^T Q_f x_T \\ \text{subject to} & x_{\tau+1} = Ax_\tau + Bu_\tau, \quad \tau = 0,\ldots,T-1 \\ & x_\tau \in \mathcal{X}, \quad \tau = 0,\ldots,T \\ & u_\tau \in \mathcal{U}, \quad \tau = 0,\ldots,T-1 \\ & x_0 = z, \end{array} \tag{3.1}$$

where $\mathcal{X} = \{x \in \mathbf{R}^n \mid x_{\min} \preceq x \preceq x_{\max}\}$ and $\mathcal{U} = \{u \in \mathbf{R}^m \mid u_{\min} \preceq u \preceq u_{\max}\}$ encapsulate lower and upper bound "box" constraints on the state and input. The variables are $x_0, \ldots, x_T \in \mathbf{R}^n$, $u_0, \ldots, u_{T-1} \in \mathbf{R}^m$, and the rest of the matrices and vectors are problem data. MPC is usually thought of as a suboptimal policy, because it involves picking

a small enough horizon T in order to make the optimization problem (3.1) tractable. The following theorem of stability for general MPC, listed for completeness, appears to be quite technical, but is really quite simple. The key takeaway is the descent condition (3.2) is the same as the descent condition (2.15).

Theorem 3 (General stability of MPC)**.** *Suppose that $f : \mathcal{X} \times \mathcal{U} \to \mathbf{R}^n$, $\ell : \mathcal{X} \times \mathcal{U} \to \mathbf{R}^n_+$, and $V_f : \mathcal{X} \to \mathbf{R}^n_+$ are continuous with $f(0,0) = 0$, $\ell(0,0) = 0$, $V_f(0) = 0$, the sets $\mathcal{X}$ and $\mathcal{X}_f$ are closed containing the origin, $\mathcal{X}_f \subseteq \mathcal{X}$ and $\mathcal{U}$ are compact containing the origin, $\mathcal{X}_f$ is control invariant for the system $x^+ = f(x,u)$, and that*

$$\min_{u \in \mathcal{U}} \{\ell(x,u) + V_f(f(x,u)) \mid f(x,u) \in \mathcal{X}_f\} \leq V_f(x), \quad \forall x \in \mathcal{X}_f,$$

then the optimal objective $V_T^\star$ of the T-horizon MPC problem can be used as a control Lyapunov function, which decreases according to

$$V_T^\star(f(x, \kappa_T(x))) - V_T^\star(x) \leq -\ell(x, \kappa_T(x)), \tag{3.2}$$

where $\kappa_T(x)$ is the MPC law.

Proof. See [RM09, §2.4] □

3.2 Box constrained quadratic programming

The MPC problem (3.1) and many others like it can be converted to a linearly constrained Quadratic Program (QP) of the form

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} x^T P x + q^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h \\ & s \succeq 0, \end{array} \tag{3.3}$$

with variables $x \in \mathbf{R}^{n_x}$ and $s \in \mathbf{R}^{n_s}$, and problem data $P = P^T$, q, A, b, G, h, and T. If $P \succeq 0$ ($P \succ 0$), then the problem (3.3) is convex (strongly convex). For example, the box constrained MPC problem (3.1) is written in the form (3.3) by using the combined variable

$x = (x_0, \ldots, x_T, u_0, \ldots, u_T) \in \mathbf{R}^{n(T+1)+mT}$ and taking

$$P = \left[\begin{array}{ccc|c|ccc} Q & \cdots & 0 & 0 & & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & & \vdots \\ 0 & \cdots & Q & & & & \\ \hline 0 & \cdots & & Q_f & & \cdots & 0 \\ \hline & & & & R & \cdots & 0 \\ \vdots & & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & & 0 & 0 & \cdots & R \end{array}\right], \quad q = \left[\begin{array}{c} 0 \\ \vdots \\ 0 \\ \hline 0 \\ \hline 0 \\ \vdots \\ 0 \end{array}\right],$$

$$A = \left[\begin{array}{cccc|c|ccccc} I & 0 & & \cdots & 0 & \cdots & & & & 0 \\ \hline A & -I & \cdots & 0 & \vdots & B & & & \cdots & 0 \\ & A & -I & \vdots & & & B & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots & & \ddots & & \\ 0 & \cdots & & A & -I & 0 & \cdots & & & B \end{array}\right], \quad b = \left[\begin{array}{c} z \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right],$$

$$G = \left[\begin{array}{cc} I_{n(T+1)\times n(T+1)} & \\ & I_{mT\times mT} \\ -I_{n(T+1)\times n(T+1)} & \\ & -I_{mT\times mT} \end{array}\right], \quad h = \left[\begin{array}{c} x_{\max} \\ \vdots \\ u_{\max} \\ \vdots \\ -x_{\min} \\ \vdots \\ -u_{\min} \\ \vdots \end{array}\right].$$

3.2.1 Primal-dual path following algorithm

Being able to solve optimization problems like (3.3) quickly and efficiently is very desirable, because such problems have to be solved over and over within the MPC loop. Finding efficient algorithms appropriate to MPC is a very active area of research. One of the most efficient algorithms involves finding the central path for the problem (3.3) by the modified

Karush–Kuhn–Tucker (KKT) conditions

$$\begin{array}{ll} \text{Stationarity:} & (Px+q)+G^Tz+A^Ty=0, \\ \text{Complementary slackness:} & z_is_i=1/t, \quad i=1,\dots,n_s \\ \text{Primal feasibility:} & Gx+s=h, \quad Ax=b, \quad s\succeq 0, \\ \text{Dual feasibility:} & z\succeq 0. \end{array} \tag{3.4}$$

The main nonlinearity in the KKT conditions is the complementary slackness condition, because it involves the product of the variables z and s. The primal-dual path following algorithm consists of updating the point $u=(x,s,z,y)$ through linearizations of the KKT conditions (3.4) and letting $t\to 0$. Specifically, we define the primal-dual residual

$$r(x,s,z,y) \triangleq \begin{bmatrix} Px+q+G^Tz+A^Ty \\ \mathbf{diag}(z)s-\frac{1}{t}\mathbf{1} \\ Gx+s-h \\ Ax-b \end{bmatrix} = \begin{bmatrix} r_x \\ r_s \\ r_z \\ r_y \end{bmatrix},$$

and approximate it by the linearization

$$r_t(u+\Delta u)\approx r_t(u)+Dr_t(u)\Delta u=0.$$

The goal at each step is to solve for Δu, where

$$Dr_t(x,s,z,y)=\begin{bmatrix} P & 0 & G^T & A^T \\ 0 & \mathbf{diag}(z) & \mathbf{diag}(s) & 0 \\ G & I & 0 & 0 \\ A & 0 & 0 & 0 \end{bmatrix}$$

is the Jacobian of $r(x,s,z,y)$. This corresponds to solving the linear system

$$\begin{bmatrix} P & 0 & G^T & A^T \\ 0 & \mathbf{diag}(z) & \mathbf{diag}(s) & 0 \\ G & I & 0 & 0 \\ A & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta s \\ \Delta z \\ \Delta y \end{bmatrix} = - \begin{bmatrix} r_x \\ r_s \\ r_z \\ r_y \end{bmatrix} \tag{3.5}$$

for $(\Delta x, \Delta s, \Delta z, \Delta y)$, updating the point (x, s, z, y), and letting $t \to \infty$. See [Van10, MB11, BV04].

3.2.2 Taking advantage of structure

By far, the most amount of time is spent solving the linear system (3.5). However, this linear system is quite sparse for MPC problems. This allows a significant amount of precomputation of the inverse to be made offline, see Figure 3.1 and [MWB11].

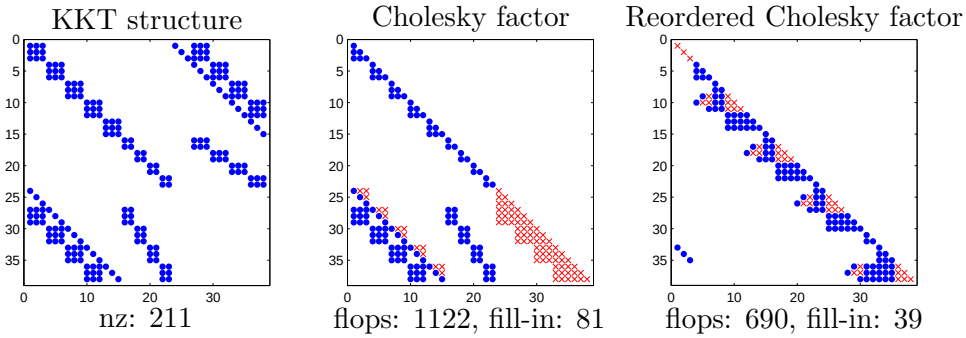


Figure 3.1: Offline structure analysis for a small MPC problem ($n_x = 3$, $n_u = 2$, $T = 4$) and a barrier method. The nonzeros of the KKT matrix are shown as circles (left). The Cholesky factor has extra fill-in nonzeros shown as crosses (middle). With the Approximate Minimum Degree (AMD) heuristic [Dav06], a reordered Cholesky factor has fewer fill-in entries (right). All these computations are performed offline, without reference to any specific MPC problem data.

3.3 Discrete-space approximate dynamic programming

3.3.1 Bounds on the value function

While MPC is a compelling suboptimal control strategy, effective implementation is limited to constrained systems for which each MPC step is expressible as a convex optimization problem. An alternative to MPC and CLF based methods is Approximate Dynamic Programming (ADP), in which we aim to approximate the Lyapunov function with the hope that the approximation will give a good control policy. While no guarantees can be made a priori that the policy is useful (or even stabilizing), experimentally these techniques can be quite effective in solving control problems that cannot be practically solved any other way.

The specific ADP strategy that forms the basis of our approach in the rest of this chapter stems from an early paper [dFR03]. Closely related was the application to hybrid systems

(mixed continuous and discrete space) found in [HR99] and the works that followed [RAN99, HR02]. This section uses notation closest to Bertsekas [BT96] to describe lower and upper bounds on the value function for discrete-space systems, and hints to why the continuous-space case is more difficult.

3.3.2 Setup

We consider a finite state Markov Decision Process. Let $\mathcal{X} = \{1, \ldots, n\}$ be a finite state space and $\mathcal{U}(i) \subseteq \mathcal{U} = \{1, \ldots, m\}$ the set of actions available in state i. In this formulation, the probability of transitioning from state i to state j under the control $u \in \mathcal{U}(i)$ is $p_{ij}(u)$, with incurred stage cost $g(i, u, j)$. Assume the transition probabilities and costs are fixed and known.

A policy is a sequence $\pi = \{\mu_0, \mu_1, \ldots\}$ where each $\mu_t : \mathcal{X} \to \mathcal{U}$ is a function that maps a state i to an available action in $\mathcal{U}(i)$. Given a policy π, the sequence of states $\{i_0, i_1, \ldots\}$ is a Markov chain with transition probabilities

$$\mathbf{P}(i_{t+1} = j \mid i_t = i) = p_{ij}(\mu_t(i)).$$

Thus, for a given policy $\pi = \{\mu_0, \mu_1, \ldots\}$, we should have

$$\sum_{j=1}^{n} p_{ij}(\mu_t(i)) = 1, \quad \text{for all } i = 1, \ldots, n.$$

The expected cost of this policy when starting from an initial state i is

$$V^{\pi}(i) = \mathbf{E}\left[\sum_{t=0}^{\infty} \gamma^t g(i_t, \mu_t(i_t), i_{t+1}) \;\middle|\; i_0 = i\right],$$

where $\gamma \in (0, 1]$ is a discount factor. For the infinite horizon case, it is often convenient to consider stationary policies $\pi = \{\mu, \mu, \ldots\}$ and $\gamma < 1$.

We can think of V^{π} as a vector in $\mathbf{R}^n$, where each component $V^{\pi}(i)$ corresponds to the expected cost-to-go starting at state i. The goal is to find a policy that minimizes the expected cost-to-go,

$$V^{\star}(i) = \min_{\pi} V^{\pi}(i).$$

3.3.3 Bellman operator

The optimal cost-to-go satisfies the Bellman equation

$$\begin{aligned} V^\star(i) &= \min_{u\in\mathcal{U}(i)} \mathbf{E}[g(i,u,j) + \gamma V^\star(j) \mid i,u] \\ &= \min_{u\in\mathcal{U}(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \gamma V^\star(j)), \quad \text{for all } i = 1,\dots,n, \end{aligned}$$

with the corresponding optimal policy at step t given by

$$\mu_t^*(i) = \operatorname*{argmin}_{u\in\mathcal{U}(i)} \mathbf{E}[g(i,u,j) + \gamma V^\star(j) \mid i,u], \quad \text{for all } i = 1,\dots,n.$$

Value iteration. For any value function vector $(V(1),\dots,V(n))$ define the vector $\mathcal{T}V$ by the Bellman operator,

$$(\mathcal{T}V)(i) = \min_{u\in\mathcal{U}(i)} \mathbf{E}[g(i,u,j) + \gamma V(j) \mid i,u],$$

thus the Bellman equation reads

$$V = \mathcal{T}V. \tag{3.6}$$

Under some regularity assumptions (*e.g.*, [BT96]) and an infinite horizon, the Bellman equation (3.6) has a unique solution $V^\star$ with a corresponding stationary policy π^*. We can arrive at the optimal value function by value iteration,

$$V^{(k+1)} = \mathcal{T}V^{(k)}, \quad k = 0,1,\dots.$$

For any starting guess $V^{(0)}$, the sequence $\{V^{(0)}, V^{(1)}, \dots\}$ converges to $V^\star$.

3.3.4 Lower and upper bounds

Monotonicity. A key property of the Bellman operator $\mathcal{T}$ is its monotonicity: if $V_1 \le V_2$, then $\mathcal{T}V_1 \le \mathcal{T}V_2$, where the inequalities are interpreted componentwise.

Lower bound. Any function V that satisfies the Bellman inequality

$$V \le \mathcal{T}V \tag{3.7}$$

automatically satisfies $V \leq V^\star$, and is a componentwise lower bound on $V^\star$. To see this, recursively apply $\mathcal{T}$ to both sides of (3.7) and use the monotonicity property,

$$V \leq \mathcal{T}V \leq \mathcal{T}^2V \leq \cdots = V^\star.$$

The Bellman inequality (3.7) defines a class of underestimators of $V^\star$, one of which is $V^\star$ itself. Members of this class capture a performance bound on the original decision problem. For example, if the transition costs are nonnegative, then $V = 0$ (componentwise) is a trivial performance bound.

Upper bound. Similarly, any function that satisfies the reverse Bellman inequality

$$\mathcal{T}V \leq V \tag{3.8}$$

automatically satisfies $V^\star \leq V$. To see this, recursively apply $\mathcal{T}$ to both sides of (3.8) and use the monotonicity property,

$$V^\star = \cdots \leq \mathcal{T}^2V \leq \mathcal{T}V \leq V.$$

Value functions that satisfy the reverse Bellman inequality (3.8) correspond to suboptimal policies, because their value is greater than or equal to the optimal value.

3.3.5 Bound optimization by linear programming

We can attempt to recover $V^\star$ by optimizing over the class of value function underestimators,

$$\begin{array}{ll} \text{maximize} & V \\ \text{subject to} & V \leq \mathcal{T}V, \end{array} \tag{3.9}$$

where the variable is the value function V, and the objective is interpreted in some scalarized way.

If the transition probabilities and stage costs are known, then we can rewrite the prob-

lem (3.9) as a linear program (LP),

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} w(i)V(i) \\ \text{subject to} & V(i) \leq \displaystyle\sum_{j=1}^{n} p_{ij}(u)(g(i,u,j)+\gamma V(j)), \\ & \qquad \forall i = 1,\ldots,n, \forall u \in \mathcal{U}(i), \end{array}$$

with variables $V(1),\ldots,V(n)$. The weights $w(1),\ldots,w(n)$ are arbitrary, as long as they are positive. The number of linear constraints is $O(nm)$.

Similarly, we can form an LP to optimize over the class of value function overestimators by minimizing instead of maximizing, and changing the sense of the inequalities. In both cases, the optimal value function $V^\star$ for the decision problem is recovered as the optimizing variable in the corresponding LP.

To make the LP useful (instead of simply solving $V = \mathcal{T}V$), we can restrict the class of underestimators by further specifying an approximating basis,

$$\widetilde{V}(i) = \sum_{k=1}^{N} \alpha_k \phi_k(i),$$

where $\phi_k : \mathcal{X} \to \mathbf{R}$ is a basis function, and α_k is a real coefficient for all $k = 1,\ldots,N$. Ideally, the number N of basis functions is less than the dimensionality n of the state space. The underapproximation LP becomes

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} w(i) \sum_{k=1}^{N} \alpha_k \phi_k(i) \\ \text{subject to} & \displaystyle\sum_{k=1}^{N} \alpha_k \phi_k(i) \leq \sum_{j=1}^{n} p_{ij}(u) \left(g(i,u,j) + \gamma \sum_{k=1}^{N} \alpha_k \phi_k(j) \right), \\ & \qquad \forall i = 1,\ldots,n, \forall u \in \mathcal{U}(i), \end{array}$$

with variables $\alpha_1,\ldots,\alpha_N$. While the number of linear constraints is still $O(nm)$, the number of variables has decreased. Note that the true value $V^\star$ is recovered if it is in the span of the basis functions, otherwise we recover a guaranteed underestimate.

3.3.6 Approximation guarantees and limitations

We wish to simultaneously find functions V^+ and V^- in an approximating class (*e.g.*, relative to a fixed basis) such that

$$V^- \leq V^\star \leq V^+,$$

and the difference between V^+ and V^- is as small as possible. To do this we might solve the LP

$$\begin{array}{ll} \text{minimize} & \max_i \{V^+(i) - V^-(i)\} \\ \text{subject to} & V^- \leq \mathcal{T}V^- \\ & \mathcal{T}V^+ \leq V^+ \\ & V^-, V^+ \in \mathcal{C} \end{array} \tag{3.10}$$

with variables V^+ and V^-. Here $\mathcal{C} \subseteq \mathbf{R}^n$ represents any, *e.g.*, basis, restrictions on the approximating class. The optimal value of (3.10), say $\epsilon^\star$, is a measure of approximation error over all states. If the overestimator and underestimator are restricted to a basis,

$$V^+(i) = \sum_{k=1}^N \alpha_k^+ \phi_k(i), \quad V^-(i) = \sum_{k=1}^N \alpha_k^- \phi_k(i),$$

then the optimization problem (3.10) has a corresponding trade-off between the number of free variables and expressiveness.

While a single-sided program like (3.9) delivers a performance bound for the decision problem, it does not relate the value function underestimate to the true value function $V^\star$. To relate to the true value function, we require both lower and upper bounds, since single-sided bounds can be arbitrarily poor, depending on the choice of basis. The double-sided program (3.10) gives a performance bound, a suboptimal policy, and a worst case approximation error, because the true value function is pointwise between the lower and upper estimates.

Nonconvexity in upper bounds. The double sided program (3.10) may not be an LP (or even convex). To see why this is the case, consider the reverse Bellman inequality

$\mathcal{T}V^+ \leq V^+$. For the finite case, it holds if and only if

$$\min_{u\in\mathcal{U}(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j)+\gamma V^+(j)) \leq V^+(i), \quad \forall i = 1,\ldots,n. \tag{3.11}$$

In other words, V^+ satisfies $\mathcal{T}V^+ \leq V^+$ if and only if for each state i there exists an input $u \in \mathcal{U}(i)$ for which the expected value is less than $V^+(i)$. The left side is not convex in the optimization variable V, because the minimum is on the wrong side of the inequality sign. This is not an issue for the Bellman inequality $V^- \leq \mathcal{T}V^-$, because it takes the form

$$V^-(i) \leq \min_{u\in\mathcal{U}(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j)+\gamma V^-(j)), \quad \forall i = 1,\ldots,n,$$

which is suitably convex. To engineer around this difficulty, we can restrict to conditions that imply the condition (3.11). For example, the conservative condition

$$\frac{1}{m}\sum_{u=1}^{m}\sum_{j=1}^{n} p_{ij}(u)(g(i,u,j)+\gamma V^+(j)) \leq V^+(i), \quad \forall i = 1,\ldots,n,$$

is suitable, because an average is always at least the minimum. (This is the condition implemented to obtain the bounds of Figure 3.4.)

3.3.7 Optimization with unknown transition probabilities

If the transition probabilities $p_{ij}(u)$ are not known ahead of time (as in an exploration or machine learning scenario), but are known to be in some set, then robust optimization can be used to design policies that are robust with respect to the uncertainty in these probabilities. As in the case of robust LP, some uncertainty descriptions are tractable.

Robust LP. Consider a linear program in inequality form,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T x \leq b_i, \quad i = 1,\ldots,m \end{array}$$

over the variable $x \in \mathbf{R}^n$, where c, b_i are fixed, and each a_i is known to lie in an ellipsoid $a_i \in \mathcal{E}_i$, where

$$\mathcal{E}_i = \{\overline{a}_i + P_i v \mid \|v\|_2 \leq 1\}.$$

The goal of a robust LP is to find a solution that works for all such $a_i \in \mathcal{E}_i$, which is equivalent to the optimization problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \sup_{a_i \in \mathcal{E}_i}(a_i^T x) \leq b_i, \quad i = 1, \ldots, m. \end{array} \tag{3.12}$$

Since the supremum in (3.12) is equal to

$$\sup_{a_i \in \mathcal{E}_i} (a_i^T x) = \overline{a}_i^T x + \|P_i^T x\|_2,$$

for every x, we can rewrite the robust LP (3.12) as the Second Order Cone Program (SOCP)

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \overline{a}_i^T x + \|P_i^T x\|_2 \leq b_i, \quad i = 1, \ldots, m. \end{array} \tag{3.13}$$

Notably, the problem (3.13) is convex, with efficient solution techniques for medium to large m and n. The additional terms $\|P_i^T x\|_2$ act as norm regularization constraints.

Robust performance bound. If the transition probabilities are known to lie in an ellipsoid, then we can rewrite the robust underapproximation program

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} w(i)V(i) \\ \text{subject to} & \displaystyle V(i) \leq \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \gamma V(j)), \\ & \qquad \forall i = 1, \ldots, n, \forall u \in \mathcal{U}(i), \forall p_{i:}(u) \in \mathcal{E}_i(u), \end{array}$$

as an SOCP. For every state i and input u, the outbound probability vector $p_{i:}(u) \in \mathcal{E}_i(u) \subseteq \mathbf{R}^n$ lies an ellipsoid, and $V \in \mathbf{R}^n$ is the variable. As a special case, this formulation can be used to find a value function (and policy) that is robust with respect to lower and upper bounds on the transition probabilities,

$$p_{ij}(u) \in [\underline{p}_{ij}(u), \overline{p}_{ij}(u)],$$

by choosing a diagonal ellipsoid that encompasses these intervals, *e.g.*,

$$\mathcal{E}_i(u) = \{\overline{a}_i(u) + P_i(u)v \mid \|v\|_2 \leq 1\}, \quad i = 1, \ldots, n, \quad u \in \mathcal{U}(i),$$

where

$$\overline{a}_i(u) = \frac{\underline{p}_{i:}(u) + \overline{p}_{i:}(u)}{2}, \quad P_i(u) = \begin{bmatrix} \frac{\overline{p}_{i1}(u) - \underline{p}_{i1}(u)}{2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\overline{p}_{in}(u) - \underline{p}_{in}(u)}{2} \end{bmatrix}.$$

3.3.8 Example: Gridworld

We construct a state space consisting of $n = 30$ states, arranged in a 5×6 grid and indexed $i = 1, \ldots, 30$ (left-right, top-bottom), modeling the movement of robot or person in a cluttered environment. At each state, there are up to four possible actions $u = 1, \ldots, 4$ corresponding to a movement N, W, S, and E, respectively. For each non-edge state i, the transition probability is defined as

$$p_{ij}(u) = \begin{cases} 0.8 & \text{if } j \text{ neighbors } i, \text{ and } u \text{ is a movement to } j, \\ 0.1/3 & \text{if } j \text{ neighbors } i, \text{ and } u \text{ is not a movement to } j, \\ 0.1 & \text{if } j = i, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for every non-edge state, there is a 0.8 chance of moving to a desired square, 0.1 chance (split evenly) to jump to one of the remaining three neighbors, and a 0.1 chance of staying at the current square. For edges and corners, we stay at the current square if a control action is unavailable.

Next, a fraction of the squares is chosen to be obstacles. These states have probability 1.0 of staying and zero probability of leaving, and are disconnected from their neighboring states.

Finally, a cost of 1 is chosen for every transition to a neighboring state, with the bottom right state (state 30) having zero cost.

We show the convergence of value iteration for two starting guesses in Figure 3.2. The next figure (Figure 3.3) shows the converged value function. In Figure 3.4, we solved a

version of the optimization problem (3.10) with $N = 10$ basis vectors. Crucially, one of the basis vectors was a vector of the form

$$\phi^{\text{obst}}(i) = \begin{cases} 1 & \text{if } i \text{ is an obstacle,} \\ 0 & \text{otherwise.} \end{cases}$$

To decrease basis complexity, we can use basis vectors that encode state membership constraints:

- A basis vector can pool over labeled, free, and obstacle regions.
- If a particular type of policy is expected, one can choose a basis vector whose gradient tends to point toward the goal.

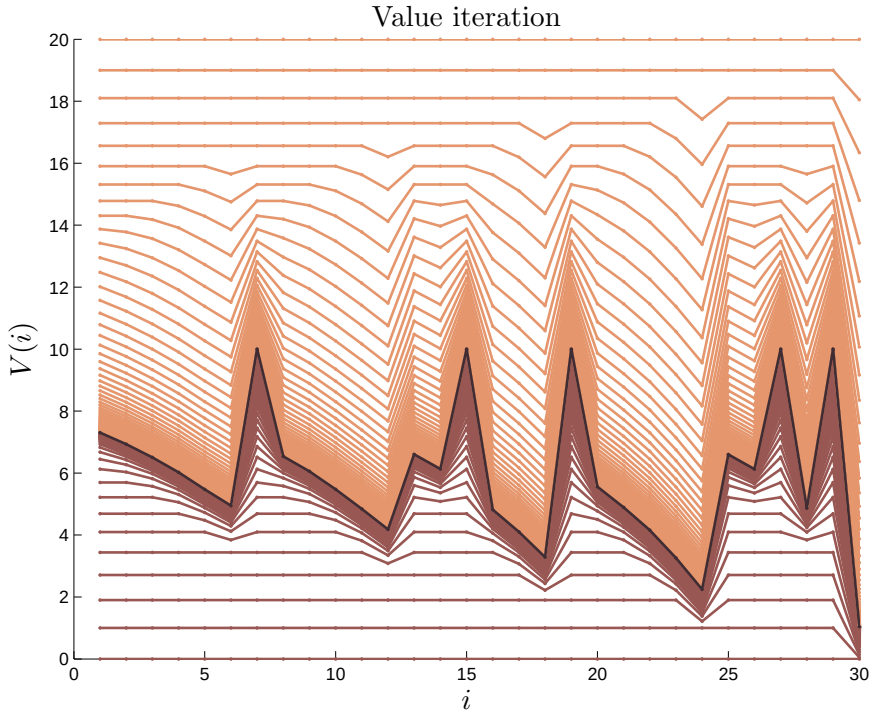


Figure 3.2: Lower and upper bounds of the value function during two instances of value iteration. Here the initial guess was $V^{(0)} = 0$ for the lower bounds and $V^{(0)} = 20$ for the upper bounds. Both iterations converge in the middle.

3.4 Continuous-space approximate dynamic programming

There are two major challenges that must be overcome when applying these techniques tractably to continuous state and action spaces, *i.e.*, $\mathcal{X} \subseteq \mathbf{R}^n$ and $\mathcal{U} \subseteq \mathbf{R}^m$. Each can

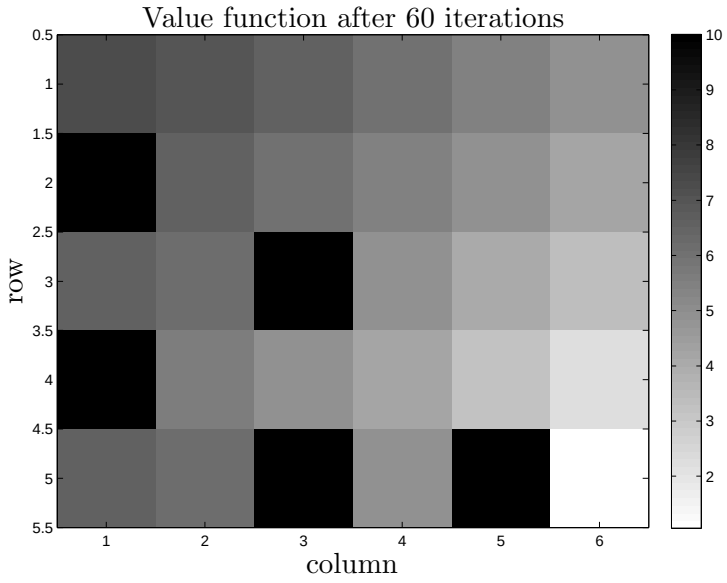


Figure 3.3: Gridworld with shading indicating the cost to go after 60 value iterations. The target (bottom-right) cell has zero cost-to-go. The obstacle states are $\mathcal{X}^{\text{obst}} = \{7, 15, 19, 27, 29\}$.

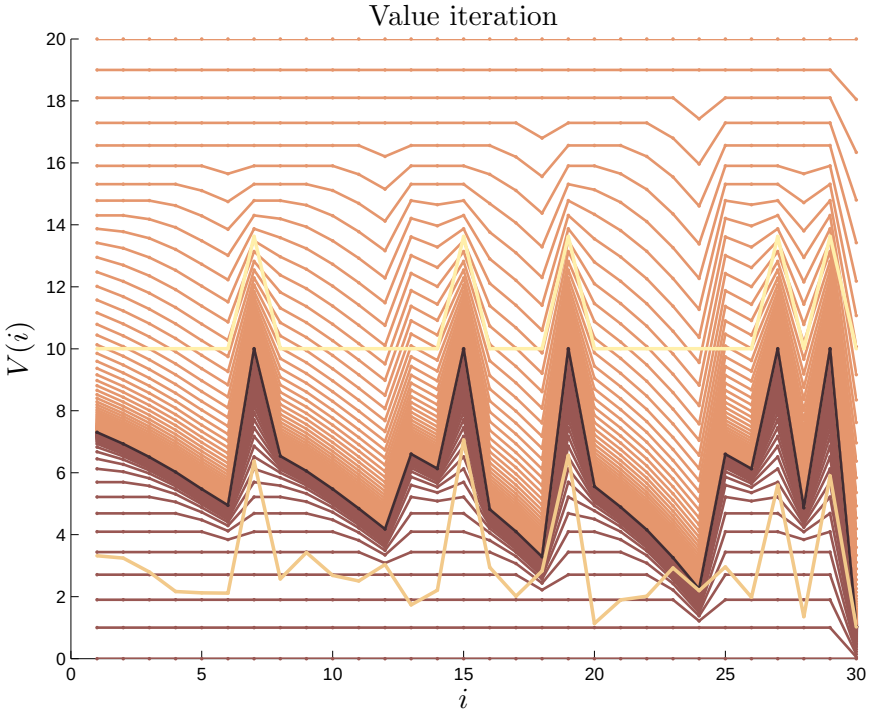


Figure 3.4: Approximate bounds using $N = 10$ basis functions, one of which is an obstacle indicator.

introduce its own level of conservativeness.

1. **Continuous state space.** The value function $V : \mathcal{X} \to \mathbf{R}$ is an infinite-dimensional object. Representing V in large dimensional states ($n \gg 1$) requires exponentially many sample points to maintain adequate detail. Typically this is partially resolved by restriction to a given basis, *e.g.*, finite elements, piecewise linear, or quadratic approximations.

2. **Continuous action space.** An implementation of the Bellman inequality (3.7) becomes a semi-infinite constraint indexed by the control input $u \in \mathcal{U}$, even when the value function is finitely parameterized. Typically, S-procedure style arguments are made here.

3.4.1 Example: Linear quadratic dynamics with bounded control

The following example from [WOB14] illustrates the type of analysis required in the continuous domain. Consider a discrete-time continuous space system

$$x_{t+1} = Ax_t + Bu_t + w_t, \quad t = 0, 1, \ldots,$$

with initial state x_0 and zero-mean noise terms $w_0, w_1, \ldots$ satisfying

$$\mathbf{E}[w_t] = 0, \quad \mathbf{E}[w_t w_t^T] = W \in \mathbf{R}^{n \times n}, \quad \text{for all } t = 0, 1, \ldots,$$

and a quadratic cost function $g(x, u) = x^T Q x + u^T R u$. The state space is unbounded, $\mathcal{X} = \mathbf{R}^n$, and the controls have an upper bound,

$$\mathcal{U} = \{u \in \mathbf{R}^m \mid \|u\|_\infty \leq 1\}.$$

Note that the main difference between this problem and a standard LQR-style problem (which has a closed form solution) is the control bound. The first choice is how to represent the infinite-dimensional value function in a finite dimensional way. We select a value function underestimator of the form

$$V(x) = x^T P x + s,$$

where $P \in \mathbf{S}^n_+$ and $s \in \mathbf{R}$ are parameters. The Bellman inequality $V \leq \mathcal{T}V$ reads

$$V(x) \leq \mathbf{E}_w[g(x,u) + \gamma V(Ax + Bu + w)], \quad \forall x \in \mathcal{X}, \forall u \in \mathcal{U}. \tag{3.14}$$

Any V satisfying (3.14) underestimates $V^\star$ over all states x. After some manipulation, this becomes

$$x^T P x + s \leq \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} \gamma A^T P A + Q & \gamma A^T P B \\ \gamma B^T P A & \gamma B^T P B + R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \gamma(\mathbf{Tr}(PW) + s), \quad \forall x \in \mathcal{X}, \forall u \in \mathcal{U}. \tag{3.15}$$

The next choice is how to implement the semi-infinite constraint (3.15). Without the control bound constraints $\|u\|_\infty \leq 1$, eq. (3.15) is a Linear Matrix Inequality (LMI) in the variables P and s. Applying the (lossy) S-procedure gives a sufficient condition for (3.15) to hold:

$$\begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} \gamma A^T P A + Q & \gamma A^T P B & 0 \\ \gamma B^T P A & \gamma B^T P B + R & 0 \\ 0 & 0 & \gamma\,\mathbf{Tr}(PW) - (1-\gamma)s \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \geq 0$$

$$\text{for all } x, u \text{ satisfying } \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & -e_i e_i^T & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \geq 0, \quad i = 1, \ldots, n.$$

This condition translates to the LMI constraint

$$\begin{bmatrix} \gamma A^T P A + Q & \gamma A^T P B & 0 \\ \gamma B^T P A & \gamma B^T P B + R + D & 0 \\ 0 & 0 & \gamma\,\mathbf{Tr}(PW) - (1-\gamma)s - \mathbf{Tr}(D) \end{bmatrix} \succeq 0,$$

where $P \succeq 0$, $s \in \mathbf{R}$, and $D \succeq 0$ are the variables, and D is restricted to be a diagonal matrix.

3.5 Extensions

With the logic specification and finite states, this discussion applies on the product state space and automaton. The engineering challenge is to pick appropriate basis vectors. With logic specification and continuous states, the value function has both continuous and discrete indices. Underestimates are more readily obtained (by convex optimization) than overestimates. Sum-of-squares can be used for overestimates. We can enforce the LP constraints only at certain specified states—more tractable with loss of bound guarantees. We discuss these issues in the next chapter.

In cases where $p_{ij}(u)$ is unknown, robust optimization can be brought to bear on this if we know something about, say, the range of probabilities $p_{ij}(u) \in [\underline{p}_{ij}(u), \overline{p}_{ij}(u)]$. We can also attempt to discover $p_{ij}(u)$, either by simulation or repeated probing. It is also possible to talk about the probability of satisfaction by incorporating this probability, directly or by proxy, into the additive stage costs. Similarly, a proxy for exploration can also be part of the objective.

3.6 References

A great book on theory and practice of MPC is [RM09]. Robust MPC is surveyed in [BM99], with robust bounding ideas going back to [BPG89, Yak92]. Using the language of upper bounds and LMIs, [KBM96] describe polytopic uncertainty in a conservative way to model nonlinear systems whose Jacobian lies in a polytope. Interior point and barrier methods for convex optimization are attributed to [NN94, Nes04], and described well by the textbook [BV04]. Specific solvers fall into three categories: general purpose solvers (SDPT3 [TTT99], SeDuMi [Stu99], CVXOPT [Van10]), parser-solvers (YALMIP [LÖ4], CVX [GB08, GB14]), and code generating solvers (CVXGEN [MB11], FORCES [DJ14]). The code generating solvers, including the central path methods described in this chapter, rely on the sparse LDL decomposition [Duf04, Dav05]. A general reference on methods for solving sparse systems is [Dav06]. General methods for ADP are described in the text [Pow07] and references therein.

Chapter 4

Automata Theory Meets Approximate Dynamic Programming

4.1 Introduction

In this chapter, we address the problem of optimal control of dynamical systems under temporal logic specifications. Dynamical systems of interest to control are typically written as differential equations on a continuous state space, with inputs that can take on a continuum of values over a continuous time interval. However, temporal logic constraints that permit decidable synthesis must work with a finite or countable parameterization of time and space. As a result, a control designer must either forgo the continuous dynamics, create a discrete abstraction, or somehow re-express the temporal constraints within their optimal control framework.

Linear Temporal Logic (LTL) is an expressive temporal logic that allows one to rigorously describe a variety of system specifications including reaching a goal, avoiding unsafe regions, covering a set of regions, responding to events, periodic surveillance, and remaining stable. To relate specifications in LTL and continuous-time continuous-state systems, researchers propose abstracting a continuous or hybrid system to a discrete, finite-state model, which allows reasoning over discrete logic properties.

Abstraction-based, hierarchical control methods have been proposed for continuous systems under temporal logic constraints [KB08, FGKGP09, HB10, LO14, AHLP00, STBR11, WTM12]. This class of methods consists of three general steps:

1. abstracting the dynamical system as a discrete finite-state system,

2. synthesizing a discrete control law using the product of the Linear Temporal Logic (LTL) specification automaton and the abstraction, and

3. refining the discrete control law and implementing it in the original system.

Approximate abstraction can be computed by reachability-based computational methods, counter-example guided abstraction refinement, and sampling-based methods [CFH$^+$03, KB08, Rei11, FT13, KF12, KSS$^+$13]. However, it is well-known that the abstraction process is computationally expensive. In addition, it cannot ensure the optimality of a control policy designed at the abstraction level with respect to a given cost function.

To address the issue of scalability, correctness, and optimality of the control design, the works [WM13, WTM13] proposed to formulate the control problem as a mixed-integer linear programming problem on the system variables, avoiding the need of a finite abstraction. However, the method has limited applicability to a fragment of LTL, and the satisfaction is encoded into a finite set of constraints over system variables corresponding to the specification. Importantly, this framework is both discrete-time and sensitive to the number of timesteps used.

Since LTL formulas can be equivalently expressed by an automaton [GO01], this chapter exploits the idea that continuous-time systems constrained by LTL can be formulated as a hybrid dynamical system by augmenting the continuous state space with the discrete states in the specification automaton. Therefore, approximate optimal control for hybrid systems can be employed to solve the optimal control under temporal logic constraints problem in a computationally efficient way.

We show how to use approximate dynamic programming (ADP) to approximate the value function and give an approximate policy. As a result, we too are limited to a subset of LTL—in this case the fragment is called co-safe LTL. An earlier similar proposed architecture using the product of a co-safe LTL specification automaton and a discrete abstraction of nonlinear robot dynamics using a sampling-based planner is [BMKV11]. By way of comparison, we treat the hybrid dynamics directly, and restrict to piecewise linear systems obviating the need for expensive RRT calculations. The main parameters under the designer's control are the value function bases used, rather than number of timesteps in the discretization or the fidelity of the discrete abstraction.

We first state the specific problem and describe the co-safe LTL fragment. We then

define the product between the continuous-time, continuous-space dynamics and the discrete automaton corresponding to the co-safe LTL specification, and write down the dynamic program corresponding to the optimal control problem. This dynamic program is difficult to solve for all but the most trivial cases, so we use the results of the previous chapter to present a novel optimization based framework for approximate dynamic programming in the temporal logic context. We also give examples of how to implement the optimization as a semidefinite program for linear systems.

4.2 Problem description

Consider a continuous-time and continuous-state dynamical system on $\mathbf{R}^n$. This system is given by

$$\dot{x} = f(x, u), \quad x(0) = x_0, \tag{4.1}$$

where $x(t) \in \mathcal{X} \subseteq \mathbf{R}^n$ and $u(t) \in \mathcal{U} \subseteq \mathbf{R}^m$ are the state and control signals at time t. For simplicity, we restrict f to be a Lipschitz continuous function of (x, u), and the control input u to be a piecewise right-continuous function of time, with finitely many discontinuities on any finite time interval. These conditions are not always required, but they ensure the existence and uniqueness of solutions, and are meant to prevent Zeno behavior.

The system (4.1) is constrained to satisfy a specification on the discrete behavior obtained from its continuous trajectory. First, let $\mathcal{AP}$ be a set of atomic propositions, which are logical predicates that hold true when $x(t)$ is in a particular region. Then, define a labeling function $L : \mathcal{X} \to \Sigma$, which maps a continuous state $x \in \mathcal{X}$ to a finite set $\Sigma = 2^{\mathcal{AP}}$ of atomic propositions that evaluate to true at x. This function partitions the continuous space $\mathcal{X}$ into regions that share the same truth values in $\mathcal{AP}$. The labeling function also links the continuous system with its *discrete behavior*. In the following definition, $\phi(x_0, [0, T], u)$ refers to the trajectory of the continuous system with initial condition x_0 under the control input $u(t)$ over the time interval $[0, T]$.

Definition. *Let $t_0, t_1, \ldots, t_N$ be times, such that*

- $0 = t_0 < t_1 < \cdots < t_N = T$,
- $L(x(t)) = L(x(t_k))$, $t_k \leq t < t_{k+1}$, $k = 0, \ldots, N$,

- $L(x(t_k^-)) \neq L(x(t_k^+))$, $k = 0, \ldots, N$.

The discrete behavior*, denoted* $\mathbf{B}(\phi(x_0, [0,T], u))$*, is defined to be the discrete word* $\sigma_0\sigma_1 \ldots \sigma_{N-1} \in \Sigma^*$*, where* $\sigma_k = L(x(t_k))$.

A specification on the discrete behavior can be written as a co-safe LTL formula over the finite set of atomic propositions (for a comprehensive description of the syntax and semantics of LTL, the reader is referred to [MP92, BK08]). A co-safe LTL formula is an LTL formula where every satisfying word has a finite good prefix[1] [KV01]. We restrict to such formulas to take advantage of the expressiveness of temporal logic for specifying optimal control problems without imposing infinite Büchi acceptance conditions. We give examples of appropriate specifications in §4.5.

Given a co-safe LTL specification φ over the set of atomic propositions $\mathcal{AP}$, there exists a corresponding deterministic finite-state automaton (DFA) $\mathcal{A}_\varphi = \langle Q, \Sigma, \delta, q_0, F\rangle$, where Q is a finite set of states (modes), $\Sigma = 2^{\mathcal{AP}}$ is a finite alphabet, $\delta : Q \times \Sigma \to Q$ is a *deterministic* transition function such that when the symbol $\sigma \in \Sigma$ is read at state q, the automaton makes a deterministic transition to state $\delta(q, \sigma) = q'$, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final, or *accepting* states. The transition function is extended to a sequence of symbols, or a *word* $w = \sigma_0\sigma_1 \ldots \in \Sigma^*$, in the usual way: $\delta(q, \sigma_0 v) = \delta(\delta(q, \sigma_0), v)$ for $\sigma_0 \in \Sigma$ and $v \in \Sigma^*$. We say that the finite word w satisfies φ if and only if $\delta(q_0, w) \in F$. The set of words satisfying φ is the *language* of the automaton $\mathcal{A}_\varphi$, denoted $\mathcal{L}(\mathcal{A}_\varphi)$.

The discrete behavior encodes the sequence of labels visited by the state as it moves along its continuous trajectory. Specifically, the atomic propositions are evaluated only at the times when the label changes value. Thus a trajectory $\phi(x_0, [0,T], u)$ satisfies an LTL specification φ if and only if its discrete behavior is in the language $\mathcal{L}(\mathcal{A}_\varphi)$. The optimal control problem is formulated as follows.

Problem. *Consider the system* (4.1)*, a co-safe LTL specification* φ*, and a final state* $x_f \in \mathcal{X}$*. Design a control law* u *that minimizes the cost function*

$$J = \int_0^T \ell(x(\tau), u(\tau))\, d\tau + \sum_{k=0}^{N} s(x(t_k), q(t_k^-), q(t_k^+)) \tag{4.2}$$

subject to the constraints that $\mathbf{B}(\phi(x_0, [0,T], u)) \in \mathcal{L}(\mathcal{A}_\varphi)$ *and* $x(T) = x_f$.

[1] Given a word $w \in \Sigma^*$ and $v \in \Sigma^*$, v is a prefix of w if and only if $w = vu$ for some $u \in \Sigma^*$. The word u is called the suffix of w.

Here, $\ell : \mathcal{X} \times \mathcal{U} \to \mathbf{R}$ is a continuous loss function, and $s : \mathcal{X} \times Q \times Q \to \mathbf{R}$ is the cost to transition between two states of the automaton whenever such a transition is allowed. The final state $x(T) = x_f$ is also specified. Similar problems have been studied in existing work [HR99, HR02, XA04, HB10, WTM12, WTM13, KSS+13]. The novelty of our approach is twofold. First, we show that this co-safe LTL problem can be cast as an optimal hybrid control problem, and second, we use approximate dynamic programming to synthesize a suboptimal controller with guaranteed performance bounds.

4.3 Product formulation

To solve Problem 4.2, we first augment the continuous state space $\mathcal{X}$ with the discrete state space Q of the specification automaton $\mathcal{A}_\varphi$ to obtain a hybrid system. The construction of $\mathcal{A}_\varphi$ for a specific formula φ can be automated with existing tools [GO01]. We show that the optimal control problem constrained by a co-safe LTL specification φ is a dynamic programming problem over the product space $\mathcal{X} \times Q$.

In this setting, the hybrid state at time t is an ordered pair $(x(t), q(t)) \in \mathcal{X} \times Q$. Evolution of the continuous-state component $x(t)$ is governed by the original system flow (4.1), while evolution of the discrete component $q(t)$ is governed by the appropriate deterministic transition of the specification automaton. Such a transition is initiated when the continuous state crosses a boundary between two labeled regions. Specifically, we consider the following product hybrid system:

Definition. *The* product system $H = \langle Q, \mathcal{X}, E, f, R, G \rangle$ *is an internally forced hybrid system, where*

- Q *is the set of discrete states (modes) of* $\mathcal{A}_\varphi$,
- $\mathcal{X} \subseteq \mathbf{R}^n$ *is the set of continuous states,*
- $E \subseteq Q \times \Sigma \times Q$ *is a set of discrete transitions, where* $e = (q, \sigma, q') \in E$ *if and only if* $\delta(q, \sigma) = q'$,
- $f : \mathcal{X} \times \mathcal{U} \to \mathbf{R}^n$ *is the continuous vector field given by* (4.1),

- $R = \{R_q \mid q \in Q\}$ *is a collection of regions, where*

$$\begin{aligned} R_{q,\sigma} &= \{x \in \mathcal{X} \mid \exists q' \in Q : (q', \sigma, q) \in E, \\ &\qquad \textit{and } L(x) = \sigma\}, \quad q \in Q, \sigma \in \Sigma, \\ R_q &= \bigcup_{\sigma \in \Sigma} R_{q,\sigma}, \quad q \in Q, \end{aligned}$$

- $G = \{G_e \mid e \in E\}$ *is a collection of guards, where*

$$G_e = \{x \in \mathbf{bd}\, R_{q,\sigma} \mid \delta(q, L(x)) = q'\},$$

for all $e = (q, \sigma, q') \in E$.

Each region R_q refers to the continuous states $x \in \mathcal{X}$ that are reachable while the automaton is in or transitions to mode q. For each discrete mode q, the continuous state evolves inside R_q until it enters a guard region $G_{(q,\sigma,q')}$ and a discrete transition to mode q' is made.

We can solve the optimal control problem with dynamic programming by ensuring that the optimal value function is zero at every accepting state of the automaton. Let $V^\star : \mathcal{X} \times Q \to \mathbf{R}$ be the optimal cost-to-go in (4.2), with $V^\star(x_0, q_0)$ denoting the optimal objective value when starting at initial condition (x_0, q_0), subject to the discrete behavior specification and final condition $x(T) = x_f$. For simplicity, we assume that $V^\star$ has no explicit dependence on t, which corresponds to searching for a stationary policy, although this assumption can be relaxed at the expense of having to choose a time-dependent basis when searching for an approximate value function later. In this setting, the cost-to-go satisfies a collection of mixed continuous-discrete Hamilton–Jacobi–Bellman (HJB) equations,

$$\begin{aligned} 0 = \min_{u \in \mathcal{U}} \left\{ \frac{\partial V^\star(x,q)}{\partial x} \cdot f(x,u) + \ell(x,u) \right\}, \\ \forall x \in R_q,\ \forall q \in Q, \end{aligned} \tag{4.3}$$

$$\begin{aligned} V^\star(x,q) = \min_{q'} \left\{ V^\star(x,q') + s(x,q,q') \right\}, \\ \forall x \in G_e,\ \forall e = (q,\sigma,q') \in E, \end{aligned} \tag{4.4}$$

$$0 = V^\star(x_f, q_f), \quad \forall q_f \in F. \tag{4.5}$$

Equation (4.3) says that $V^\star(x, q)$ is an optimal cost-to-go inside the regions where the label remains constant. The next equation (4.4) is a shortest-path equality that must hold at every continuous state x where discrete state transition to a different label can happen. Finally, the boundary equation (4.5) fixes the value function.

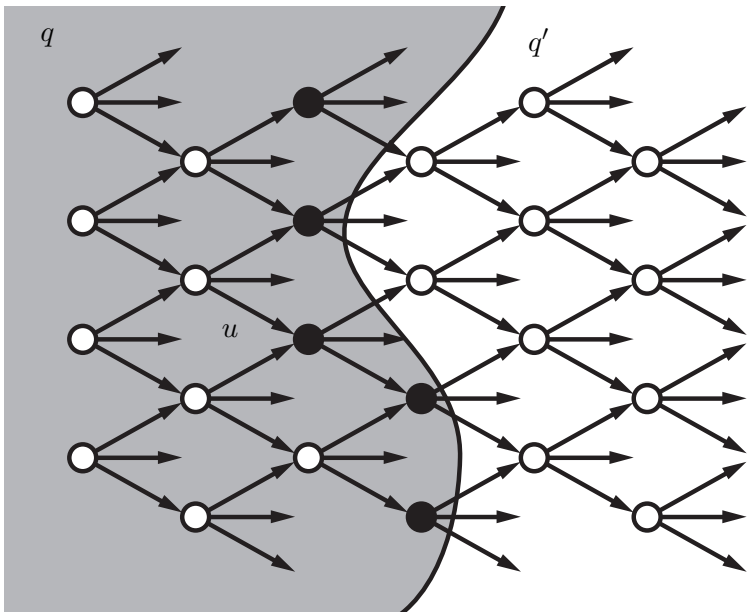


Figure 4.1: Finite state interpretation of HJB conditions (4.3)–(4.5)

We can interpret these HJB conditions intuitively as a single-sink shortest-path problem on a directed weighted graph, where nodes with the same label are treated together and the weights are the incremental costs $\ell(x, u)dt$ or the discrete transition costs $s(x, q, q')$ (Figure 4.1). As long as the continuous state evolves within the same labeled region, the value function is subject to the optimality condition associated with the region that contains that state. As a result, the continuous-state condition (4.3) must hold on the interior nodes (white), while the discrete-state switching condition (4.4) must hold at the boundary nodes (black).

The graph interpretation also clarifies why automata derived from co-safe LTL specifications fit within the dynamic programming framework but not automata derived from more general temporal logics: the semantics of general LTL are over infinite execution traces, and require Büchi automata whose acceptance conditions do not readily translate to a single-sink shortest-path problem. Nevertheless, we believe the co-safe restriction is a strength, rather than weakness, because co-safe LTL is still highly expressive, and the

solution methods we describe in the next section are efficient for many classes of problems, relatively simple to implement, and can be readily automated.

4.4 Lower bounds on the optimal cost

Let $V^\star$ be a value function satisfying the hybrid HJB conditions (4.3)–(4.5), and suppose V is another function that satisfies the following inequalities,

$$\begin{aligned} 0 \leq & \frac{\partial V(x,q)}{\partial x} \cdot f(x,u) + \ell(x,u), \\ & \forall x \in R_q,\ \forall u \in \mathcal{U},\ \forall q \in Q, \end{aligned} \tag{4.6}$$

$$\begin{aligned} 0 \leq & V(x,q') - V(x,q) + s(x,q,q'), \\ & \forall x \in G_e,\ \forall e = (q,\sigma,q') \in E, \end{aligned} \tag{4.7}$$

$$0 = V(x_f, q_f), \quad \forall q_f \in F. \tag{4.8}$$

Then $V(x_0,q_0) \leq V^\star(x_0,q_0)$. This approach is motivated by [HR99, Ran99, dFR03]. The inequalities (4.6)–(4.8) characterize a set of optimal value function under-estimators, among which is the optimal value function $V^\star$ itself. The difference between the equalities (4.3)–(4.5) and the inequalities (4.6)–(4.8) is the removal of the minimum operators in favor of semi-infinite constraints and the addition of pointwise inequalities.

The strength of using inequalities to search over value function under-estimators, instead of solving the HJB equations directly, lies in an ability to come up with *approximate* value functions and ADP policies whose suboptimality can be quantified [WB09]. The ADP method is enabled by the fact that we can come up with sufficient conditions that imply (4.6)–(4.8), and relies on finding the largest approximate value function that is a pointwise underestimate of $V^\star$. Thus we solve the problem

$$\begin{array}{ll} \text{maximize} & V(x_0,q_0) \\ \text{subject to} & (4.6), (4.7), \text{ and } (4.8) \end{array} \tag{4.9}$$

over the variables parameterizing V.

The approximate value function V is written as a sum of basis functions as follows,

$$V(x,q) = \sum_{i=1}^{n_q} w_{i,q}\phi_{i,q}(x),$$

where $\phi_{i,q} : \mathcal{X} \to \mathbf{R}$ are given basis functions, and the coefficients $w_{i,q}$, $i = 1, \ldots, n_q$ are the variables. Given an approximate value function V, an approximately optimal control law can be implemented as

$$u(x,q) = \operatorname*{argmin}_{u \in \mathcal{U}} \left\{ \frac{\partial V(x,q)}{\partial x} \cdot f(x,u) + \ell(x,u) \right\}.$$

Switching between different discrete modes is autonomous, and the scaling with problem size can be controlled with an appropriate parameterization of V. For more information on ADP, see the references [BT96, POW07]. The goal of the rest of this section is to describe how to solve (4.9) for specific problems.

4.4.1 Linear quadratic systems

In this section we describe how to search for an approximate value function for the linear system

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0,$$

with a quadratic continuous cost

$$\ell(x,u) = x^T Q x + u^T R u, \quad Q \succeq 0, \quad R \succ 0,$$

and a constant switching cost

$$s(x,q,q') = \begin{cases} \xi & \text{if } q \neq q', \\ 0 & \text{otherwise,} \end{cases}$$

where $\xi > 0$ is a given positive constant.

We parameterize the approximate value function as a quadratic. For each $q \in Q$, let

$$V(x,q) = x^T P_q x + 2r_q^T x + t_q, \quad \text{for all } x \in \mathcal{X}, \tag{4.10}$$

where $P_q = P_q^T \in \mathbf{R}^{n\times n}$, $r_q \in \mathbf{R}^n$ and $t_q \in \mathbf{R}$ are the variables of the parameterization. This parameterization is linear in (P_q, r_q, t_q) and corresponds to choosing basis functions $\phi_{i,q}(x)$ that are quadratic, linear, and constant in the components of x, respectively.

With the approximate value function in (4.10), the objective function is then

$$V(x_0, q_0) = x_0^T P_{q_0} x_0 + 2r_{q_0}^T x_0 + t_{q_0}.$$

For the constraints, condition (4.6) is the same as

$$0 \leq \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} A^T P_q + P_q A + Q & P_q B & A^T r_q \\ B^T P_q & R & B^T r_q \\ r_q^T A & r_q^T B & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \qquad \forall x \in R_q,\ \forall u \in \mathcal{U},\ \forall q \in Q, \tag{4.11}$$

i.e., it is a collection of $|Q|$ semi-infinite constraints indexed by the continuous variables $x \in \mathcal{X}$ and $u \in \mathcal{U}$, one for each $q \in Q$. If R_q is a quadratically representable set, *e.g.*, ellipsoids or halfspaces, then we directly use the S-procedure to obtain a finite number of sufficient conditions for (4.11) to hold [WB09]. Otherwise if R_q is not quadratically representable, then two approaches can be used. One approach is exact: Given state $q \in Q$, we partition R_q to a finite set of quadratically representable sets $R_q = \bigcup_{i=1}^{N_q} R_q^i$ and enforce constraint (4.11) in state q to each R_q^i with parameterization (P_q^i, r_q^i, t_q^i). Note that such a partition is possible if the set of states satisfying each atomic proposition is quadratically representable. Alternatively, we can overapproximate R_q with a quadratically representable set. In both cases, $\mathcal{U}$ can be overapproximated by a quadratically representable set.

Similarly, the inequality (4.7) is enforced over each guard region by the collection of $|E|$ semi-infinite constraints

$$0 \leq \begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} P_{q'} - P_q & r_{q'} - r_q \\ r_{q'}^T - r_q^T & t_{q'} - t_q + \xi \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} \qquad \forall x \in G_e,\ \forall e \in E. \tag{4.12}$$

Once again, the S-procedure translates the semi-infinite constraints into a finite sufficient condition if the guard regions are quadratically representable. The guard regions are

quadratically representable when each atomic proposition corresponds to a quadratically representable set in $\mathcal{X}$.

Finally, the condition (4.8) relates P_{q_f}, r_{q_f}, and t_{q_f} via the $|F|$ linear equality constraints

$$0 = x_f^T P_{q_f} x_f + 2 r_{q_f}^T x_f + t_{q_f}, \quad \forall q_f \in F, \tag{4.13}$$

where x_f is the given fixed final state.

4.4.2 Nonlinear systems

For general nonlinear flows, it is appropriate to choose a more expressive value function approximation. A typical approach would use a radial basis function (RBF) basis [PS91], [LW13, §12],

$$V(x,q) = \sum_{i=1}^{m} w_{i,q} \exp\left(-\frac{\|x - c_i\|_2^2}{a_i^2}\right), \tag{4.14}$$

where $\{c_i\}_{i=1}^m$ is a finite set of center points in $\mathbf{R}^n$ chosen to sample the continuous state space $\mathcal{X}$, and $\{a_i\}_{i=1}^m$ are positive constants that define the RBF widths. The same set of basis functions can be used everywhere, or alternatively the RBF centers can be chosen to have most of their support over the regions R_q appropriate to each state $q \in Q$.

With the approximate value function in (4.14), the objective and constraints (4.6)–(4.8) lead to the optimization problem

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{m} w_{i,q} \phi_{i,q}(x_0) \\
\text{subject to} \quad & 0 \le \sum_{i=1}^{m} w_{i,q} \left(\frac{\partial \phi_{i,q}(x)}{\partial x} \cdot f(x,u)\right) + \ell(x,u) \\
& \qquad \forall x \in R_q,\ \forall u \in \mathcal{U},\ \forall q \in Q, \\
& 0 \le \sum_{i=1}^{m} (w_{i,q'} - w_{i,q}) \phi_i(x) + s(x,q,q') \\
& \qquad \forall x \in G_e,\ \forall e = (q,\sigma,q') \in E, \\
& 0 = \sum_{i=1}^{m} w_{i,q_f} \phi_i(x_f) \quad \forall q_f \in F,
\end{aligned}$$

with variables $w_{i,q}$. The switching cost function s can be defined similar to that in §4.4.1.

This semi-infinite LP has a finite number $m \times |Q|$ of variables but an infinite number of constraints. A general solution approach is to sample the constraints, and then solve the

finite linear program. The derivation of Probably Approximately Correct (PAC) bounds on the needed number of samples requires careful study of the specific dynamics and costs in the optimization problem.

4.5 Examples

4.5.1 Linear quadratic systems with halfspace labels

We consider the linear quadratic system on $\mathcal{X} = \mathbf{R}^2$ from §4.4.1 with the specific parameters

$$A = \begin{bmatrix} 2 & -2 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$
$$Q = I, \quad R = 1, \quad \xi = 1,$$
$$x_0 = (0.5, 0), \quad x_f = (0, 0).$$

Let $\mathcal{AP} = \{a, b\}$ consist of atomic propositions that are true whenever the continuous state enters a specific region,

$$a = \text{True} \iff (x(t) \in R_A), \quad R_A = \{x \in \mathbf{R}^2 \mid x_1 \leq 1\}$$
$$b = \text{True} \iff (x(t) \in R_B), \quad R_B = \{x \in \mathbf{R}^2 \mid x_1 > 1\}.$$

Note that R_A and R_B are closed halfspaces and the interface between them is the line $G = \{x \in \mathbf{R}^2 \mid x_1 = 1\}$. We define the labeling function $L : \mathbf{R}^2 \to \{A, B\} \subseteq \Sigma = 2^{\mathcal{AP}}$, where $A = \{a\}$ and $B = \{b\}$. The goal is to satisfy the specification

$$\varphi_1 = \Diamond A \wedge \Diamond B,$$

where $\Diamond$ is the LTL "eventually" operator.

The automaton that accepts this specification consists of four states and is shown in Figure 4.2. The discrete behavior accepted by this automaton is any trajectory that eventually visits each of the regions R_A and R_B.

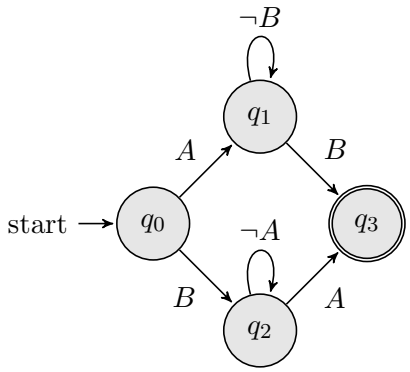


Figure 4.2: Automaton $\mathcal{A}_{\varphi_1}$ for $\varphi_1 = \Diamond A \wedge \Diamond B$

The guard regions are defined as

$$G_{(q_0,A,q_1)} = R_A, \quad G_{(q_0,B,q_2)} = R_B,$$
$$G_{(q_1,B,q_2)} = G_{(q_2,A,q_2)} = G.$$

Note that the direction of crossing the guard region is encoded by the allowed DFA transitions.

Guided by the automaton, we mechanically apply the S-procedure to obtain the semidefinite program

$$\begin{aligned} \text{maximize} \quad & x_0^T P_{q_0} x_0 + 2 r_{q_0}^T x_0 + t_{q_0} \\ \text{subject to} \quad & (4.11), (4.12), \text{ and } (4.13), \end{aligned}$$

with variables $P_q = P_q^T \in \mathbf{R}^{2\times 2}$, $r_q \in \mathbf{R}^2$, $t_q \in \mathbf{R}$, $q \in Q = \{q_0, \ldots, q_3\}$, and eight additional variables coming from the S-procedure. With $x_f = 0$, the final constraint (4.13) translates to $t_{q_3} = 0$.

The semidefinite program was solved using SDPT3 and resulted, within numerical ac-

curacy, in the following value function:

$$P_q^\star = \begin{bmatrix} 22.314 & -28.142 \\ -28.142 & 38.799 \end{bmatrix}, \quad q = q_0, \ldots, q_3,$$

$$r_q^\star = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad q = q_0, \ldots, q_3,$$

$$t_q^\star = \begin{cases} 2, & q = q_0 \\ 1, & q = q_1, q_2 \\ 0, & q = q_3. \end{cases}$$

Note that the shape of $V(\cdot, q)$ is the same for every state in the automaton, with the only difference being the offset t_q. The policy implied by this value function is illustrated in Figure 4.3 for the specific hybrid execution trace with initial condition $x_0 = (0.5, 0)$. The reader is invited to follow the figure as we interpret the execution:

1. The path $x(t)$ starts at the point a with initial condition $x_0 = (0.5, 0)$ and automaton state q_0.

2. The automaton makes an immediate transition to q_1, because $L(x_0) = A$ and the value function is lower for this discrete state. The continuous dynamics follow the negative gradient of $V(\cdot, q_1)$.

3. At point b, the automaton transitions to q_3. The continuous dynamics go down the gradient of $V(\cdot, q_3)$ in the segment of the path between b and c.

4. At point c, the automaton is already in its accepting state q_3. The continuous dynamics continue to follow the negative gradient of $V(\cdot, q_3)$ to reach $x_f = 0$.

4.5.2 More complex specification

We now consider three regions, R_A, R_B, and R_C with the slightly more complex specification

$$\varphi = (A \to \Diamond B) \wedge (C \to \Diamond B) \wedge (\Diamond A \vee \Diamond C).$$

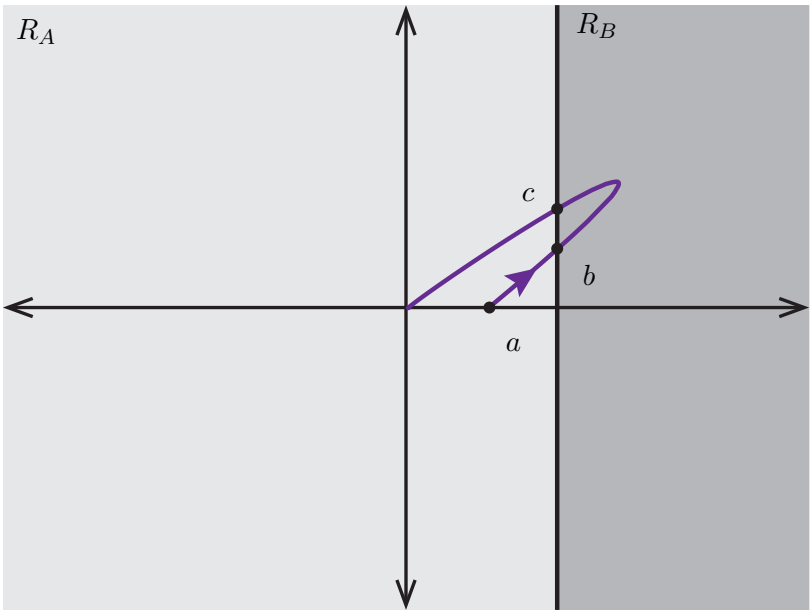


Figure 4.3: Approximately minimum cost path satisfying φ_1 with initial condition $x_0 = (0.5, 0)$. To satisfy φ_1, the trajectory must leave R_A and visit R_B.

This specification ensures that either R_A or R_C must be reached, after which the system must eventually visit R_B. The automaton for this specification is shown in Figure 4.4. Depending on the accrued continuous and transition costs, there is a choice to go left or right in Figure 4.5.

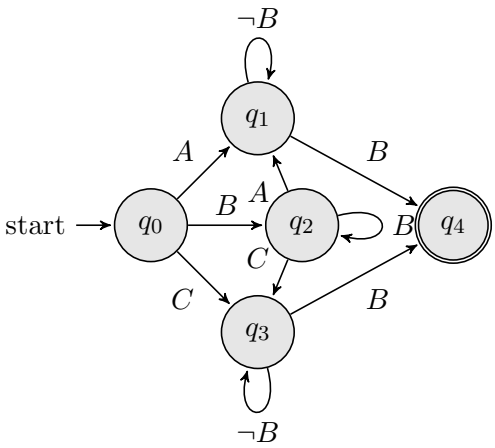


Figure 4.4: Automaton $\mathcal{A}_{\varphi_2}$ for $\varphi_2 = (A \to \Diamond B) \wedge (C \to \Diamond B) \wedge (\Diamond A \vee \Diamond C)$

We form the semidefinite program as before to obtain five approximate value functions $V(\cdot, q)$, one for each $q \in Q = \{q_0, \ldots, q_4\}$ in the automaton. This time, we plot the execution for two initial conditions $x_0 = (-0.5, -0.5)$, whose path (abc) goes right, and $x_0 = (-0.5, 0)$,

whose path (def) goes left. See Figure 4.5.

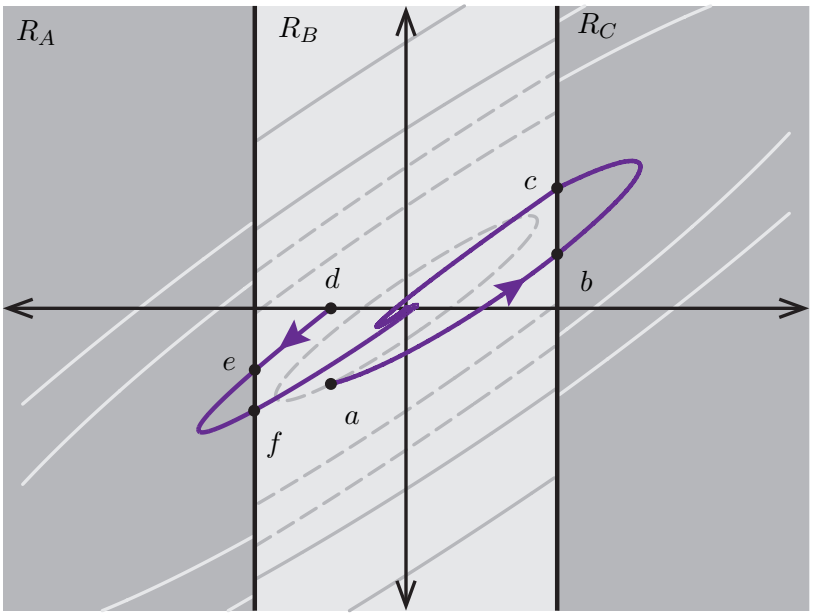


Figure 4.5: Approximately minimum cost paths satisfying φ_2, and level sets of the value function active in each region: the path abc with initial condition $x_0 = (-0.5, -0.5)$ satisfies φ_2 by visiting R_C, while the path def with initial condition $x_0 = (-0.5, 0)$ satisfies φ_2 by visiting R_A. Note that the level sets of $V(\cdot, q_2)$ (solid, inside R_B) have a subtle tilt and magnitude shift compared to $V(\cdot, q_4)$ (dashed, inside R_B), which allows for the excursion away from the origin required by φ_2.

To interpret this policy, it is valuable to compare the spectra of the closed loop matrix

$$A_q^{\text{cl}} = A - BR^{-1}B^T P_q^\star$$

in the initial mode $q = q_0$ against the accepting mode $q = q_4$,

$$\lambda(A_{q_0}^{\text{cl}}) = \{0.786 \pm 1.144i\}, \quad \lambda(A_{q_4}^{\text{cl}}) = \{-1 \pm i\}.$$

In the initial state q_0, the closed loop eigenvalues are unstable, while they are stable in the final state q_4. Our procedure therefore recovers the requirement of φ_2 that a trajectory starting near the origin in region R_B must go away to visit another region, and eventually transition to an accepting state of the automaton before being allowed back to $x_f = 0$.

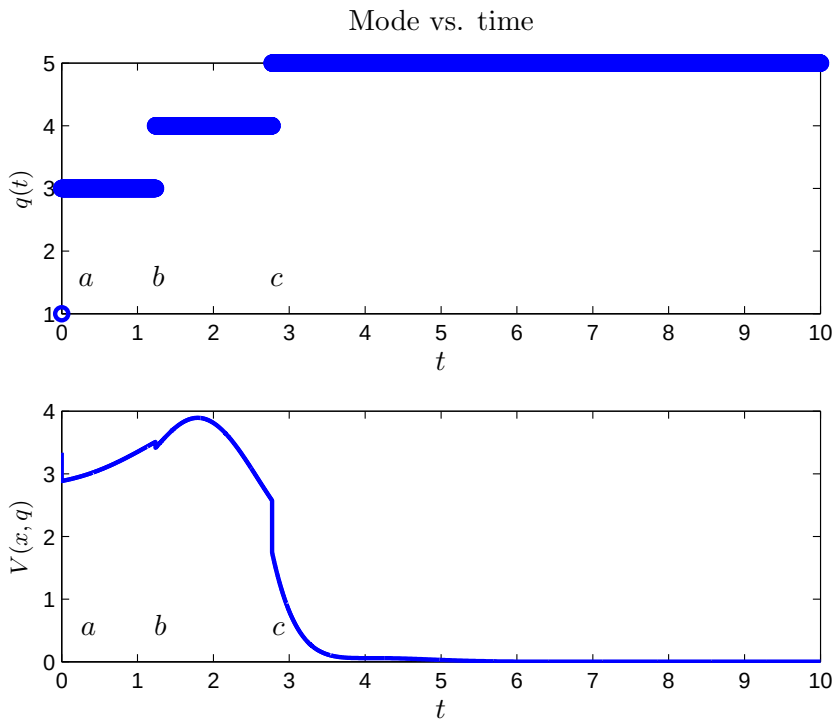


Figure 4.6: State of $\mathcal{A}_{\varphi_2}$ and value function along the path abc going right.

4.6 Conclusion

This chapter presented an approach to formulate and solve the optimal control problem under co-safe LTL constraints using approximate dynamic programming. The optimal policy is given by following a sequence of value functions over a hybrid state space, where the continuous component comes from the continuous-time and continuous-state dynamics of the system, and the discrete component comes from the specification automaton. For linear dynamics and quadratic-constant costs, we showed how to use the specification automaton to construct a semidefinite program that gives a suboptimal policy. This procedure does not rely on discretizing the time/state space or formulating non-convex optimization problems. At this stage, this approach is limited to a subset of LTL specifications that admit deterministic and finite (rather than Büchi) automaton representations. Extensions to the general class of LTL specifications that admit deterministic Büchi automaton representations with continuous-time dynamics are subjects of current work.

Future work will also include the investigation of the PAC bound on sampling-based methods for nonlinear systems. The proposed framework can also be incorporated as a

building block in other approximate optimal control methods for scalable synthesis of systems with LTL specifications.

Chapter 5

Adaptation and Learning

5.1 Introduction

In this chapter, we bridge the Lyapunov bound thinking of the previous chapters with the well-developed theory of adaptive control. Specifically, we review its concepts and make connections to machine learning systems. Fundamentally, we view adaptive control not just as the result of a clever zeroing of dissipation terms in a Lyapunov argument, but rather as the implementation of a specific (continuous-time or discrete-time) algorithm for solving a constrained convex optimization problem online. While much of this material is quite standard, the constrained optimization lens through which we view adaptive control is critical to understanding the upcoming Chapter 6.

We motivate the discussion by the following example of a Model Referenced Adaptive Controller (MRAC).

Example 5.1 Let $A \in \mathbf{R}^{n\times n}$ be a dynamics matrix, and $B \in \mathbf{R}^{n\times m}$ with (A, B) controllable. Consider the state space system

$$\dot{x} = Ax + B(u + c), \quad x(0) = x_0, \tag{5.1}$$

where $x \in \mathbf{R}^n$ is the state, $u \in \mathbf{R}^m$ is the input, and $c \in \mathbf{R}^m$ is an unknown but constant disturbance vector satisfying $Bc \neq 0$. Let $Q \succ 0$ and $R \succ 0$ be given matrices that define a linear state feedback controller $u_{\mathrm{LQR}} = Kx$, where $K = -R^{-1}B^TP$, and $P \succ 0$ satisfies the algebraic Riccati equation

$$A^TP + PA - PBR^{-1}B^TP + Q = 0.$$

It can be shown that setting $u = u_{\text{LQR}}$ in (5.1) gives

$$\limsup_{t\to\infty} \|x(t)\| = \| - (A+BK)^{-1}Bc\| \neq 0.$$

In other words, the LQR controller fails to stabilize the system (5.1). However, setting $u = u_{\text{ideal}} = u_{\text{LQR}} - c$ in (5.1) would asymptotically stabilize the system, because the closed loop matrix $A + BK$ is stable from LQR theory.

MRAC controller. Since the ideal controller is not practical (it requires knowing c), we will instead consider $u_{\text{MRAC}} = u_{\text{LQR}} - \hat{c}(t)$, where $\hat{c}(t)$ is a possibly time-varying estimate of c. Our goal is to come up with an estimation rule for $\hat{c}(t)$ that ensures asymptotic convergence of the state in (5.1) to zero. To do this, we will pick a stable (model) reference system

$$\dot{x}^{\text{ref}} = (A+BK)x^{\text{ref}}, \quad x^{\text{ref}}(0) = x_0$$

and drive the state $x(t)$ toward $x_{\text{ref}}(t)$ with u_{MRAC}. The following estimation rule will form the basis of our analysis:

$$\dot{\hat{c}}(t) = \Gamma B^T P e(t), \quad \hat{c}(0) = \hat{c}_0, \tag{5.2}$$

where $e(t) = x(t) - x^{\text{ref}}(t)$, $\Gamma = \Gamma^T \succ 0$ is a fixed $m \times m$ matrix called the *learning rate*, and $\hat{c}_0 \in \mathbf{R}^m$ is a given initial guess.

With $u = u_{\text{MRAC}}$, we have the following system and reference dynamics

$$\begin{aligned} \dot{x} &= (A+BK)x - B(\hat{c}(t) - c) \\ \dot{x}_{\text{ref}} &= (A+BK)x_{\text{ref}}. \end{aligned}$$

Subtracting these gives the model tracking error dynamics

$$\dot{e} = (A+BK)e - B\tilde{c}(t),$$

and since c is a constant, $\dot{\tilde{c}}(t) = \dot{\hat{c}}(t) = \Gamma B^T P e(t)$, we can summarize the dynamics of $(e, \tilde{c})$ as

$$\frac{d}{dt}\begin{bmatrix} e \\ \tilde{c} \end{bmatrix} = \begin{bmatrix} A+BK & -B \\ \Gamma B^T P & 0 \end{bmatrix}\begin{bmatrix} e \\ \tilde{c} \end{bmatrix}, \quad \begin{bmatrix} e(0) \\ \tilde{c}(0) \end{bmatrix} = \begin{bmatrix} 0 \\ \hat{c}_0 - c \end{bmatrix}. \tag{5.3}$$

Bounded estimates. We use the Lyapunov function candidate

$$V : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}, \quad V(e, \tilde{c}) = e^T P e + \tilde{c}^T \Gamma^{-1} \tilde{c},$$

where $\tilde{c}(t) = \hat{c}(t) - c$, to show that $e(t)$ and $\tilde{c}(t)$ are bounded for all $t \geq 0$. To see this, note that V is positive definite, and

$$\begin{aligned}
\dot{V}(e, \tilde{c}) &= 2e^T P \dot{e} + 2\tilde{c}^T \Gamma^{-1} \dot{\tilde{c}} \\
&= 2e^T P((A + BK)e - B\tilde{c}) + 2\tilde{c}^T \Gamma^{-1}(\Gamma B^T P e) \\
&= e^T (\underbrace{(A + BK)^T P + P(A + BK)}_{= A^T P + PA - 2PBR^{-1}B^T P})e + 2\tilde{c}^T (\underbrace{B^T P e - B^T P e}_{=0}) \\
&= -e^T (Q + PBR^{-1}B^T P)e \leq -e^T Q e,
\end{aligned}$$

where the last equality follows from the algebraic Riccati equation. Thus $\dot{V} \leq 0$. This immediately implies that $e(t)$ and $\tilde{c}(t)$ are bounded in magnitude for all time $t \geq 0$, because

$$\begin{aligned}
V(e(t), \tilde{c}(t)) &= V(e(0), \tilde{c}(0)) + \int_0^t \underbrace{\dot{V}(e(\tau), \tilde{c}(\tau))}_{\leq 0} \, d\tau \\
&\leq V(e(0), \tilde{c}(0)).
\end{aligned}$$

However, their boundedness does not by itself show that the signals $e(t)$ and $\tilde{c}(t)$ tend to zero. To conclude something about the asymptotic value of $e(t)$ and $\tilde{c}(t)$, we use the following lemma:

Lemma (Barbalat)**.** *Let $f : [0, \infty) \to \mathbf{R}$ be a uniformly continuous function and suppose that $\lim_{t\to\infty} \int_0^t f(\tau)\, d\tau$ exists and is finite. Then $f(t) \to 0$ as $t \to \infty$.*

To apply Barbalat's lemma to $\dot{V}$, we first show that $\dot{V}$ is uniformly continuous:

$$\begin{aligned}
\ddot{V}(e, \tilde{c}) &= -2e(t)^T (Q + PBR^{-1}B^T P)\dot{e}(t) \\
&= -2e(t)^T (Q + PBR^{-1}B^T P)((A + BK)e(t) - B\tilde{c}(t)).
\end{aligned}$$

Since $e(t)$ and $\tilde{c}(t)$ are bounded, so is $\ddot{V}$. Furthermore,

$$\int_0^T \dot{V}(e(t), \tilde{c}(t))\, dt = \underbrace{V(e(T), \tilde{c}(T))}_{\text{bounded for all } T} - V(e(0), \tilde{c}(0)),$$

therefore $\dot{V}(e, \tilde{c}) \to 0$ by Barbalat's lemma. Since $\dot{V}$ is strictly negative definite in the $e(t)$ component, this in turn implies $e(t) \to 0$. Finally, since $e(t) = x(t) - x^{\text{ref}}(t)$ goes to zero, and $x^{\text{ref}}(t)$ goes to zero, the state $x(t)$ in (5.1) goes to zero, as required.

Persistence of excitation. It is still possible that $\tilde{c}(t) \not\to 0$ even though $e(t) \to 0$. In other words, the estimate $\hat{c}(t)$ need not converge to the true value c, even though the state is driven to zero. For the parameter estimate to be driven to the true estimate, the system may have to satisfy a further *persistence of excitation* condition. See [IF06] and Chapter 6.

Link to integral control. For comparison, consider the controller $u_{\text{int}} = u_{\text{LQR}} - q(t)$, where

$$q(t) = \Gamma B^T P \int_0^t x(\tau)\, d\tau,$$

and $\Gamma = \Gamma^T \succ 0$ is a fixed gain matrix. In other words, u_{int} is a state feedback LQR controller with an additional integral term. With $u = u_{\text{int}}$, we have

$$\frac{d}{dt}\begin{bmatrix} x \\ q \end{bmatrix} = \underbrace{\begin{bmatrix} A + BK & -B \\ \Gamma B^T P & 0 \end{bmatrix}}_{A_{\text{cl}}} \begin{bmatrix} x \\ q \end{bmatrix} + \begin{bmatrix} Bc \\ 0 \end{bmatrix}, \tag{5.4}$$

which looks just like the error dynamics of the MRAC (5.3), except with a nonzero driving term.

If the system (5.4) had a steady state value and the relevant inverse existed, then the steady state would be given by

$$\begin{bmatrix} x_{\text{ss}} \\ q_{\text{ss}} \end{bmatrix} = -\begin{bmatrix} A + BK & -B \\ \Gamma B^T P & 0 \end{bmatrix}^{-1} \begin{bmatrix} Bc \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ c \end{bmatrix}.$$

In particular, $x(t) \to x_{\text{ss}} = 0$, as required. (We can check that this is indeed a solution by multiplying both sides by A_{cl}.) In fact, these dynamics are the same as the MRAC when

we view them in terms of the variables $x(t)$ and $\tilde{q}(t) = q(t) - c$:

$$\frac{d}{dt}\begin{bmatrix} x \\ \tilde{q} \end{bmatrix} = \begin{bmatrix} A + BK & -B \\ \Gamma B^T P & 0 \end{bmatrix} \begin{bmatrix} x \\ \tilde{q} \end{bmatrix}.$$

As we've seen, these dynamics ensure $x(t) \to 0$. A very famous property of integral controllers that they can reject constant disturbances, as is the case here. Thought of another way, an integral controller is a zero-referenced MRAC for linear systems.

5.2 Robust adaptation

The MRAC is not robust with respect to bounded process noise without further modification. Consider a zero-referenced, robust SISO MRAC design problem with the following open-loop plant model,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + b_u(u(t) - \theta^T x(t)) + b_w w(t) \\ x(0) &= x_0, \end{aligned}$$

reference model

$$\begin{aligned} \dot{x}^{\text{ref}}(t) &= Ax^{\text{ref}}(t) \\ x^{\text{ref}}(0) &= x_0^{\text{ref}}, \end{aligned}$$

and tracking error $e(t) = x(t) - x^{\text{ref}}(t)$. The tracking error dynamics are

$$\begin{aligned} \dot{e}(t) &= \dot{x}(t) - \dot{x}^{\text{ref}}(t) \\ &= Ax(t) + b_u(u(t) - \theta^T x(t)) + b_w w(t) - Ax^{\text{ref}} \\ &= Ae(t) + b_u(u(t) - \theta^T x(t)) + b_w w(t). \end{aligned}$$

Assuming full state feedback, we use the control input $u(t) = \hat{\theta}(t)^T x(t)$, where $\hat{\theta}(t)$ is to be determined. Define $\tilde{\theta}(t) = \hat{\theta}(t) - \theta$ as the parameter error. With this definition the tracking error obeys the dynamics

$$\dot{e}(t) = Ae(t) + b_u\tilde{\theta}(t)^T x(t) + b_w w(t). \tag{5.5}$$

Let $V(e,\tilde{\theta}) = e^TPe + \tilde{\theta}^T\Gamma^{-1}\tilde{\theta}$ be a Lyapunov function candidate. The time derivative along the trajectories of (5.5) is

$$\begin{aligned}\dot{V}(e,\tilde{\theta}) &= (Ae + b_u\tilde{\theta}^Tx + b_ww)^TPe + e^TP(Ae + b_u\tilde{\theta}^Tx + b_ww) + \dot{\tilde{\theta}}^T\Gamma^{-1}\tilde{\theta} + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} \\ &= \begin{bmatrix} e \\ w \\ \tilde{\theta} \end{bmatrix}^T \begin{bmatrix} A^TP + PA & Pb_w & 0 \\ b_w^TP & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e \\ w \\ \tilde{\theta} \end{bmatrix} + 2\tilde{\theta}^T(\Gamma^{-1}\dot{\tilde{\theta}} + xe^TPb_u). \end{aligned} \tag{5.6}$$

A typical approach in adaptive control is to pick $\dot{\tilde{\theta}}$ (or equivalently the parameter update law $\dot{\hat{\theta}} = \dot{\tilde{\theta}} + \dot{\theta} = \dot{\tilde{\theta}}$ because θ is a constant) so the second term is zero, *i.e.*,

$$\dot{\tilde{\theta}} = -\Gamma xe^TPb_u, \tag{5.7}$$

however as we will see, the adaptive law (5.7) can lead to runaway parameters.

Suppose $\|w(t)\|_2 \leq w_{\max}$ for all t. With the adaptive law (5.7), the Lyapunov function $\dot{V}(e,\tilde{\theta}) < 0$ only on the dissipation set

$$\mathcal{D} = \left\{ (e,\tilde{\theta}) \neq (0,0) \;\middle|\; \begin{bmatrix} e \\ w \\ \tilde{\theta} \end{bmatrix}^T \begin{bmatrix} A^TP + PA & Pb_w & 0 \\ b_w^TP & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e \\ w \\ \tilde{\theta} \end{bmatrix} < 0,\ \|w\|_2 \leq w_{\max} \right\} \subseteq \mathbf{R}^n \times \mathbf{R}^n.$$

Thus, while the tracking error $e(t)$ will be ultimately bounded,

$$\|e(t)\|_2 \leq 2\frac{\lambda_{\max}(P)}{\lambda_{\min}(Q)}\|b_w\|_2 w_{\max} \triangleq e_0, \quad \text{for all } t \geq T_0,$$

where $A^TP + PA \triangleq -Q \prec 0$, the parameter error $\tilde{\theta}(t)$ need not be bounded. Even though the system might be tracking the reference model, the parameter estimate $\hat{\theta}(t)$ might tend to infinity.

As a simple fix, one can modify (5.7) to ensure that $(e,\tilde{\theta}) \in \mathcal{D}$ by freezing the evolution of $\hat{\theta}(t)$ when the tracking error is small. The *deadzone modification* does this by setting the

parameter derivative to zero,

$$\dot{\hat{\theta}} = \begin{cases} -\Gamma x e^T P b_u & \text{if } \|e\| > e_0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.8}$$

While the deadzone modification solves the problem of runaway parameters, the modification still does not ensure parameter convergence, $\tilde{\theta}(t) \to 0$, by itself.

One courageous attempt to combat the problem of unbounded parameters due to nonzero noise might be to bound $\|H_{\tilde{\theta}w}(s)\|_\infty \leq \sqrt{\gamma}$ by imposing the dissipation condition

$$\dot{V}(e,\tilde{\theta}) + \tilde{\theta}^T\tilde{\theta} - \gamma w^T w \leq 0, \quad \text{for all } w^T w \leq w_{\max}^2,$$

which by the S-procedure leads to the following LMI in the variables P, τ, γ

$$P \succ 0, \quad \tau, \gamma \geq 0, \quad \begin{bmatrix} A^TP + PA & Pb_w & 0 & 0 \\ b_w^TP & -\gamma I & 0 & 0 \\ 0 & 0 & (1-\tau)I & 0 \\ 0 & 0 & 0 & \tau w_{\max}^2 \end{bmatrix} \preceq 0. \tag{5.9}$$

However, it is clear that (5.9) is infeasible when $w_{\max} \neq 0$, because the bottom-right block

$$\begin{bmatrix} (1-\tau)I & 0 \\ 0 & \tau w_{\max}^2 \end{bmatrix}$$

cannot be made negative semidefinite for any $\tau \geq 0$. The presence of a nonzero disturbance really does have a dangerous, unbounded effect on the parameter estimates.

Directly incorporating robustness. While failing to guarantee runaway parameters due to bounded disturbances, the adaptation law (5.7) can still be designed to be robust to bounded-energy $w(t)$. For example, the w-to-e induced $\mathbf{L}_2$ gain is bounded by $\sqrt{\gamma}$, provided we can ensure $\dot{V}(e,\tilde{\theta}) + e^Te - \gamma w^T w \leq 0$ for all (e,w) satisfying (5.5) and (5.7), or equivalently, the following LMI condition holds

$$P \succ 0, \quad \begin{bmatrix} A^TP + PA + I & Pb_w \\ b_w^TP & -\gamma I \end{bmatrix} \preceq 0.$$

Thus, we can minimize $\|H_{ew}(s)\|_\infty$ of the adaptive laws (5.7) or (5.8) by solving the semidefinite program

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & P \succ 0 \\ & \begin{bmatrix} A^TP + PA + I & Pb_w \\ b_w^TP & -\gamma I \end{bmatrix} \preceq 0 \end{array} \tag{5.10}$$

over the variables P and γ. By the bounded-real lemma, the optimum value of (5.10) is given by $\gamma^\star = \|(sI - A)^{-1}b_w\|_2^2$. Hence, with an appropriate choice of P, a bounded-energy disturbance $w(t)$ leads to a bounded-energy reference tracking error.

Passivation. We can try to passivate the w-to-$\tilde{\theta}$ response by imposing the condition

$$\dot{V}(e, \tilde{\theta}) + \sigma\tilde{\theta}^T\tilde{\theta} - \tilde{\theta}^T b_w w \leq 0,$$

which would in principle be accomplished by the adaptive law

$$\dot{\tilde{\theta}} = -\Gamma x e^T P b_u - \sigma\tilde{\theta} + b_w w. \tag{5.11}$$

However, the adaptive law (5.11) can only be implemented under the conditions that $\tilde{\theta}$ and $b_w w$ are available for measurement. While it may be possible to measure the model noise $b_w w$ when implementing (5.11) online, the parameter error $\tilde{\theta}$ would only be known if the true parameter θ were known. Thus the passivation idea is not implementable. Instead, the law (5.11) is often rewritten as

$$\dot{\hat{\theta}} = -\Gamma x e^T P b_u - \sigma\hat{\theta},$$

which is implementable. While this *σ-modification* solves the problem of runaway parameters, its main issue is the tendency of the parameter estimate $\hat{\theta}(t)$ to tend to zero, meaning the adaptive controller "unlearns" the parameters after a while.

General additive modifications. We can consider the effect of an entire suite of additive modifications to the standard adaptive law (5.7) by examining the second term of the Lyapunov derivative (5.6). Note that any additive modification must enter the Lyapunov

derivative with the term $\tilde{\theta}^T$ in tow, meaning that any feedback with the signals e, w, and $\tilde{\theta}$ would appear as $\tilde{\theta}^T e$, $\tilde{\theta}^T w$, and $\tilde{\theta}^T\tilde{\theta}$. Members of this suite include the σ-modification, e-modification, and the projection operator, see [NA05, IF06, LW13].

Robust reference model. Suppose that $A \in \mathbf{Co}\{A_1, \ldots, A_L\}$, which corresponds to a reference model that differs from the original dynamics. A polytopic model uncertainty leads to the adaptive law (5.5), from which the parameter matrix P can be chosen by solving

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & P \succ 0 \\ & \begin{bmatrix} A_i^T P + P A_i + I & P b_w \\ b_w^T P & -\gamma I \end{bmatrix} \preceq 0, \quad i = 1, \ldots, L, \end{array}$$

over the variables P and γ.

5.3 Dynamic estimation

5.3.1 Least squares estimator

Consider the following process

$$\begin{cases} y_t = \theta^T \phi(x_t) \\ \hat{y}_t = \hat{\theta}_t^T \phi(x_t), \end{cases}$$

where $e_t = \hat{y}_t - y_t \in \mathbf{R}$ is the estimation error, and $\hat{\theta}_t \in \mathbf{R}^n$ is the estimate of an unknown constant $\theta \in \mathbf{R}^n$ at time t. The time index t can be discrete or continuous, but having a continuous time index simplifies the treatment and is more in line with adaptive estimation on a fast scale, *cf.* [IF06].

A typical estimation objective might be to minimize the running cost

$$J(\hat{\theta}_t) = \frac{1}{2} \int_0^t \left(\hat{\theta}_t^T \phi(x_\tau) - y_\tau \right)^2 d\tau, \tag{5.12}$$

which is the cost of making a constant estimate $\hat{\theta}_t$ over the entire interval $[0, t]$. Thus at each time t, we solve the (unconstrained) least-squares optimization problem

$$\text{minimize} \quad J(\hat{\theta}_t) \tag{5.13}$$

over the variable $\hat{\theta}_t \in \mathbf{R}^n$. It is important to note that even in the presence of the integral in (5.12), the optimization problem (5.13) is convex and finite dimensional. Its solution in closed form is found by setting the gradient of $J(\hat{\theta}_t)$ to zero,

$$\nabla_{\hat{\theta}_t} J(\hat{\theta}_t) = \int_0^t \phi(x_\tau) \left(\hat{\theta}_t^T \phi(x_\tau) - y_\tau \right) \, d\tau \stackrel{\text{set}}{=} 0,$$

or equivalently the optimality condition is

$$\left(\int_0^t \phi(x_\tau) \phi(x_\tau)^T \, d\tau \right) \hat{\theta}_t = \int_0^t \phi(x_\tau) y_\tau \, d\tau. \tag{5.14}$$

Thus the optimal estimate at time t is

$$\hat{\theta}_t = \underbrace{\left(\int_0^t \phi(x_\tau) \phi(x_\tau)^T \, d\tau \right)^{-1}}_{\Gamma_t} \int_0^t \phi(x_\tau) y_\tau \, d\tau, \tag{5.15}$$

provided the matrix Γ_t is invertible, $\Gamma_t \succ 0$ (which is implied by persistence of excitation). In order to compute $\hat{\theta}_t$ in (5.15), it appears that we need to calculate two integrals and to invert a matrix. However, these computations can be significantly simplified by making two observations.

First, we differentiate the identity $\Gamma_t \Gamma_t^{-1} = I$ to yield $\dot{\Gamma}_t \Gamma_t^{-1} + \Gamma_t (\dot{\Gamma_t^{-1}}) = 0$, giving

$$\begin{aligned} \dot{\Gamma}_t &= -\Gamma_t (\dot{\Gamma_t^{-1}}) \Gamma_t \\ &= -\Gamma_t \phi(x_t) \phi(x_t)^T \Gamma_t, \end{aligned}$$

and second we differentiate the optimality condition (5.14) to get

$$\phi(x_t) \phi(x_t)^T \hat{\theta}_t + \Gamma_t^{-1} \dot{\hat{\theta}}_t = \phi(x_t) y_t,$$

which results in

$$\begin{aligned} \dot{\hat{\theta}}_t &= -\Gamma_t \left(\phi(x_t) \phi(x_t)^T \hat{\theta}_t - \phi(x_t) y_t \right) \\ &= -\Gamma_t \phi(x_t) (\hat{y}_t - y_t) \\ &= -\Gamma_t \phi(x_t) e_t, \end{aligned}$$

where we made use of the identity $\hat{y}_t = \phi(x_t)^T\hat{\theta}_t = \hat{\theta}_t^T\phi(x_t)$. To summarize the construction, the value $\hat{\theta}_t$ in (5.15) can equivalently be found by integrating the following set of differential equations

$$\begin{cases} \dot{\Gamma}_t = -\Gamma_t\phi(x_t)\phi(x_t)^T\Gamma_t \\ \dot{\hat{\theta}}_t = -\Gamma_t\phi(x_t)e_t, \end{cases} \tag{5.16}$$

up to time t, with some initial conditions $\Gamma_0 \in \mathbf{S}^n_{++}$ and $\hat{\theta}_0 \in \mathbf{R}^n$.

The rule (5.16) is implementable, assuming state feedback: the model prediction error $e_t = \hat{y}_t - y_t$ and the state regressor $\phi(x_t)$ drive the parameter estimate dynamics, while the learning rate Γ_t is also changed in accordance with the measurements. Note that the differential equations (5.16) are nonlinear in $(\Gamma_t, \hat{\theta}_t)$, and correspond to classical "MIT-rule" gradient estimation if the learning rate Γ_t is frozen in time.

5.3.2 Constrained estimator

If we know that the unknown parameter θ must be in a convex set $\Theta \subseteq \mathbf{R}^n$, we can solve the constrained least-squares problem

$$\begin{array}{ll} \text{minimize} & J(\hat{\theta}_t) \\ \text{subject to} & \hat{\theta}_t \in \Theta. \end{array}$$

As long as the objective J is a convex function of $\hat{\theta}_t$, and the constraint set Θ is a convex set, the optimization problem can be efficiently solved. For example, in discrete time, a simple projected gradient descent estimator would take the form

$$\hat{\theta}_{t+1} := \Pi_\Theta\left(\hat{\theta}_t - \Gamma_t\nabla_{\hat{\theta}_t}J(\hat{\theta}_t)\right),$$

where $\Pi_\Theta(x) = \operatorname{argmin}_{z\in\Theta}\|z - x\|_2$ is the Euclidean projection on the set Θ. In many cases, both in discrete and continuous time, the estimation problem can be implemented recursively, or as the result of a Newton-style interior point iteration. In the case of recursive estimation, this technique is sometimes called *recursive least squares*, or constrained *Kalman filtering*. In the case of interior point iteration, it is known as Model Predictive or Moving Horizon Estimation. The next section describes a specific case where Θ is an affine set.

5.3.3 Equality constrained least squares estimator

If we know that the unknown parameter θ belongs to a subspace, say $h^T\theta = c$ with a given vector $h \in \mathbf{R}^n$ and real constant c, we can constrain the estimate $\hat{\theta}_t$ to also lie in that subspace. Thus at each time t, we can solve the constrained optimization problem

$$\begin{array}{ll} \text{minimize} & J(\hat{\theta}_t) \\ \text{subject to} & h^T\hat{\theta}_t = c \end{array} \tag{5.17}$$

over the variable $\hat{\theta}_t$. Despite the appearance of integrals, the optimization problem (5.17) is finite dimensional and convex.

After introducing a Lagrange multiplier $\nu_t \in \mathbf{R}$, the KKT conditions read

$$\begin{bmatrix} P_t & h \\ h^T & 0 \end{bmatrix} \begin{bmatrix} \hat{\theta}_t \\ \nu_t \end{bmatrix} = \begin{bmatrix} g_t \\ c \end{bmatrix}, \tag{5.18}$$

where

$$P_t = \int_0^t \phi(x_\tau)\phi(x_\tau)^T \, d\tau, \quad g_t = \int_0^t \phi(x_\tau) y_\tau \, d\tau.$$

We arrive at the recursive version of this estimator by differentiating (5.18) with respect to time,

$$\begin{bmatrix} \phi(x_t)\phi(x_t)^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{\theta}_t \\ \nu_t \end{bmatrix} + \begin{bmatrix} P_t & h \\ h^T & 0 \end{bmatrix} \begin{bmatrix} \dot{\hat{\theta}}_t \\ \dot{\nu}_t \end{bmatrix} = \begin{bmatrix} \phi(x_t) y_t \\ 0 \end{bmatrix},$$

or equivalently

$$\begin{bmatrix} P_t & h \\ h^T & 0 \end{bmatrix} \begin{bmatrix} \dot{\hat{\theta}}_t \\ \dot{\nu}_t \end{bmatrix} = \begin{bmatrix} -\phi(x_t)(\hat{y}_t - y_t) \\ 0 \end{bmatrix}.$$

Note that a block LDU (Aitken) diagonalization of the KKT matrix is given by

$$\begin{bmatrix} P_t & h \\ h^T & 0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ h^T P_t^{-1} & 1 \end{bmatrix} \begin{bmatrix} P_t & 0 \\ 0 & -h^T P_t^{-1} h \end{bmatrix} \begin{bmatrix} I & P_t^{-1} h \\ 0 & 1 \end{bmatrix},$$

hence we can compute the KKT inverse as

$$\begin{aligned}
\begin{bmatrix} P_t & h \\ h^T & 0 \end{bmatrix}^{-1} &= \begin{bmatrix} I & P_t^{-1}h \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} P_t & 0 \\ 0 & -h^T P_t^{-1} h \end{bmatrix}^{-1} \begin{bmatrix} I & 0 \\ h^T P_t^{-1} & 1 \end{bmatrix}^{-1} \\
&= \begin{bmatrix} I & -P_t^{-1}h \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_t^{-1} & 0 \\ 0 & -1/h^T P_t^{-1} h \end{bmatrix} \begin{bmatrix} I & 0 \\ -h^T P_t^{-1} & 1 \end{bmatrix} \\
&= \begin{bmatrix} I & -P_t^{-1}h \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_t^{-1} & 0 \\ (h^T P_t^{-1} h)^{-1} h^T P_t^{-1} & -(h^T P_t^{-1} h)^{-1} \end{bmatrix} \\
&= \begin{bmatrix} P_t^{-1} - P_t^{-1}h(h^T P_t^{-1} h)^{-1} h^T P_t^{-1} & P_t^{-1}h(h^T P_t^{-1} h)^{-1} \\ (h^T P_t^{-1} h)^{-1} h^T P_t^{-1} & -(h^T P_t^{-1} h)^{-1} \end{bmatrix} \\
&= \bar{\Gamma}_t,
\end{aligned}$$

meaning that

$$\begin{aligned}
\dot{\bar{\Gamma}}_t &= -\bar{\Gamma}_t (\dot{\bar{\Gamma}_t^{-1}}) \bar{\Gamma}_t \\
&= -\bar{\Gamma}_t \begin{bmatrix} \dot{P}_t & 0 \\ 0 & 0 \end{bmatrix} \bar{\Gamma}_t.
\end{aligned}$$

Equivalently, we may write

$$\begin{cases}
\begin{bmatrix} \dot{\Gamma}_t & \dot{\gamma}_t \\ \dot{\gamma}_t^T & \dot{\xi}_t \end{bmatrix} := - \begin{bmatrix} \Gamma_t \\ \gamma_t^T \end{bmatrix} \phi(x_t)\phi(x_t)^T \begin{bmatrix} \Gamma_t & \gamma_t \end{bmatrix} \\
\begin{bmatrix} \dot{\hat{\theta}}_t \\ \dot{\nu}_t \end{bmatrix} := - \begin{bmatrix} \Gamma_t \\ \gamma_t \end{bmatrix} \phi(x_t)(\hat{y}_t - y_t).
\end{cases}$$

5.3.4 Learning example: Finite measure estimation

Suppose θ denotes a finite probability measure, with the constraints $\mathbf{1}^T\theta = 1$ and $\theta_i \geq 0$. The constrained least squares algorithm is then driven by the estimation difference

$$\begin{aligned}
e_t &= \hat{y}_t - y_t \\
&= \hat{\theta}_t^T \phi(x_t) - \theta^T \phi(x_t) \\
&= \mathbf{E}_{\hat{\theta}_t}[\phi(x_t)] - \mathbf{E}_{\theta}[\phi(x_t)],
\end{aligned}$$

i.e., the statistical regret, or dispersion.

A simple example concerns the following autonomous (discrete-time) system,

$$x_{t+1} = A_t x_t, \quad t = 0, 1, 2, \ldots, \tag{5.19}$$

where $x_0 \in \mathbf{R}^n$ is given, and A_t is a matrix randomly chosen at each time t from the finite set $\{A^{(1)}, \ldots, A^{(L)}\}$. The choice probabilities

$$\theta_j = \mathbf{P}(A_t = A^{(j)}), \quad j = 1, \ldots, L,$$

are unknown. The goal of the problem is to determine the distribution θ online.

Instances of adaptive algorithms for determining θ take an interesting form. For simplicity, we discuss the gradient estimator. Suppose with the specific regressors

$$\Phi(x)_{ij} = (A^{(j)}x)_i, \quad i = 1, \ldots, n, \quad j = 1, \ldots, L,$$

we can measure the expected next-step location

$$\begin{aligned}(y_t)_i &= \sum_{j=1}^{L} \theta_j \Phi(x_t)_{ij} = \sum_{j=1}^{L} \theta_j (A^{(j)} x_t)_i \\ &= (\mathbf{E}_\theta[A_t x_t])_i, \quad i = 1, \ldots, n.\end{aligned}$$

This can be accomplished, for example, by sampling N points $x_{t+1}^{(1)}, \ldots, x_{t+1}^{(N)}$ from a simulation oracle and forming the empirical average

$$\mathbf{E}_\theta[A_t x_t] \approx \frac{1}{N} \sum_{k=1}^{N} x_{t+1}^{(k)}.$$

The normalized gradient descent algorithm takes the following instantiation,

$$\begin{cases} e_t := \mathbf{E}_{\hat{\theta}_t}[A_t x_t] - \mathbf{E}_\theta[A_t x_t] \\ \hat{\theta}_{t+1} := \dfrac{1}{1 - \mathbf{1}^T \Gamma \Phi(x_t) e_t} (\hat{\theta}_t - \Gamma \Phi(x_t) e_t), \end{cases} \tag{5.20}$$

provided it is initialized with a valid distribution $\hat{\theta}_0$ (*e.g.*, the unform distribution $\hat{\theta}_0 = (1/L)\mathbf{1}$) and the learning rate is small enough that each iterate $\hat{\theta}_t$ is also a distribution.

At each time step t, the first expectation $\mathbf{E}_{\hat{\theta}_t}[A_t x_t]$ is a function of the current parameter estimate $\hat{\theta}_t$ and state measurement x_t, while $\mathbf{E}_\theta[A_t x_t]$ is the answer to N queries of a simulation oracle. In the extreme case of $N = 1$, the algorithm (5.20) is a version of *stochastic gradient descent*.

It can be shown using a Chernoff bound that if the process (5.19) is almost surely persistently exciting (PE),

$$\exists T, \alpha > 0, \quad \frac{1}{T}\sum_{\tau=t}^{t+T} \phi(x_\tau)\phi(x_\tau)^T \succeq \alpha I, \quad \text{for all } t, \quad \text{(a.s.)}, \tag{5.21}$$

then θ is Probably Approximately Correct (PAC) learnable, meaning that

$$\mathbf{P}(\|\hat{\theta}_t - \theta\|_2 > \epsilon) \leq \delta, \quad \text{if} \quad t > \frac{C(\alpha)}{2N\epsilon^2} \log \frac{2}{\delta}.$$

In other words, the distribution θ can be learned within an arbitrarily small tail probability δ to any accuracy ϵ, provided the algorithm has been running long enough. For a review of concepts in learnability, see, *e.g.*, [Val84, Kak03].

It is interesting to examine the PE condition (5.21). The process (5.19) might fail to satisfy (5.21) by not exciting all the relevant "statistical" or "measurement" modes. For example, if all $A^{(i)}$ have a common unobservable subspace with respect to the measurement $y_t = \phi(x_t)$, then the measurement modes are not excited, and we cannot expect to determine the distribution θ. While every classically PE measurement process is also almost surely PE, an interesting topic of future research is to determine if there exist useful almost surely PE processes that are not classically PE. If they did, then some level of robustness and ergodicity could be exploited in *e.g.*, network applications. Further implications of weakened PE conditions are discussed in Chapter 6. Still, even if the process is not almost surely PE, the estimation error e_t tends to zero (a.s.) by the same Lyapunov arguments.

5.4 Case study: Controlling wing rock

In this section we design a controller to robustly stabilize the "wing rock phenomenon" in a delta-wing aircraft at a high angle of attack. Such aircraft are known to be unstable in the roll direction at high angle of attack, therefore active control must be used to avoid the

instability. At a given trim condition, the wing roll model is nonlinear,

$$\begin{aligned}\dot{\varphi} &= p \\ \dot{p} &= \theta_1\varphi + \theta_2 p + (\theta_3|\varphi| + \theta_4|p|)p + \theta_5\varphi^3 + \theta_6\delta_a,\end{aligned}$$

where φ is the roll angle (rad), p is the roll rate (rad/s), and δ_a is the differential aileron control (rad). The state is (φ, p) and the input is δ_a. We use the same parameters as in [LW13, EX. 9.3],

$$\theta_1 = -0.018, \quad \theta_2 = 0.015, \quad \theta_3 = -0.062, \quad \theta_4 = 0.009, \quad \theta_5 = 0.021, \quad \theta_6 = 0.75.$$

We can rewrite the dynamics in parameter-affine form

$$\underbrace{\begin{bmatrix}\dot{\varphi}\\ \dot{p}\end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix}0 & 1\\ \theta_1 & \theta_2\end{bmatrix}}_{A}\underbrace{\begin{bmatrix}\varphi\\ p\end{bmatrix}}_{x} + \underbrace{\begin{bmatrix}0\\ \theta_6\end{bmatrix}}_{B}\left(\underbrace{\delta_a}_{u} + \underbrace{\frac{1}{\theta_6}\begin{bmatrix}\theta_3 & \theta_4 & \theta_5\end{bmatrix}}_{\theta^T}\underbrace{\begin{bmatrix}|\varphi|p\\ |p|p\\ \varphi^3\end{bmatrix}}_{\phi(x)}\right). \tag{5.22}$$

The reference roll dynamics are given by a second order system

$$\underbrace{\begin{bmatrix}\dot{\varphi}^{\mathrm{ref}}\\ \dot{p}^{\mathrm{ref}}\end{bmatrix}}_{\dot{x}^{\mathrm{ref}}} = \underbrace{\begin{bmatrix}0 & 1\\ -\omega_n^2 & -2\xi\omega_n\end{bmatrix}}_{A^{\mathrm{ref}}}\underbrace{\begin{bmatrix}\varphi^{\mathrm{ref}}\\ p^{\mathrm{ref}}\end{bmatrix}}_{x^{\mathrm{ref}}} + \underbrace{\begin{bmatrix}0\\ \omega_n^2\end{bmatrix}}_{B^{\mathrm{ref}}}\underbrace{\varphi^{\mathrm{cmd}}}_{r}, \tag{5.23}$$

where $\varphi^{\mathrm{cmd}}(t)$ is the reference roll angle, $\omega_n = 1$ (rad/s) is the natural frequency, and $\xi = 0.7$ is the damping ratio. The goal is to design a control input $u = \delta_a$ that makes the roll dynamics (5.22) approach the reference dynamics (5.23).

5.4.1 Open loop limit cycle

With $\delta_a = 0$, the roll dynamics (5.22) are unstable. Specifically, trajectories starting near the origin reach a limit cycle in (φ, p) space, as seen in Figure 5.1. The cross-coupling leads to bank angle magnitudes in excess of 30°.

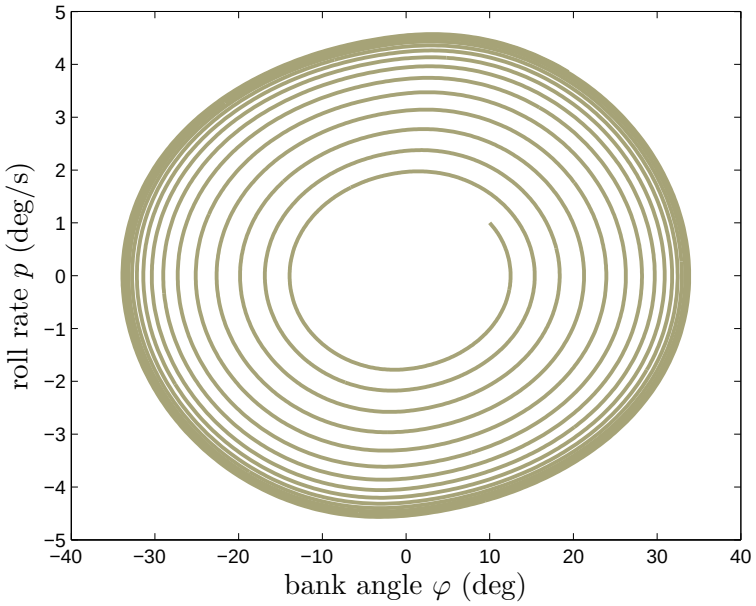


Figure 5.1: The open-loop delta wing dynamics exhibit a wing rock instability.

5.4.2 Robust design

For simplicity, we assume that θ_1, θ_2, and θ_6 are known. However, θ_3, θ_4, and θ_5 are only known within $\pm 20\%$. Thus we have

$$\theta \in \mathbf{Co}\left\{(-0.062/0.75 \cdot (1 \pm 0.2), 0.009/0.75 \cdot (1 \pm 0.2), 0.021/0.75 \cdot (1 \pm 0.2))\right\}.$$

See Figures 5.2–5.4.

5.5 References

Adaptive control dates back to the 1960s MIT rule, and has a storied past, with a great historical overview given by Anderson [And05] and Stein [Ste03]. As detailed in these historical accounts, the "robust" modifications like the deadzone [PN82], σ-mod [IK83], e-mod [NA87], and projection operator [KN82, PP92] are critical for safe adaptive methods, because unmodeled or unmatched noise can often be detrimental to closed loop stability. General references are [NA05, IF06, LW13]. Constraints on input in adaptive methods were studied by [LH07]. LMI based approaches also appear in [PKP09, YYCS09]. Furthermore, nonquadratic Lyapunov functions were studied in [ERA05]. Early attempts at

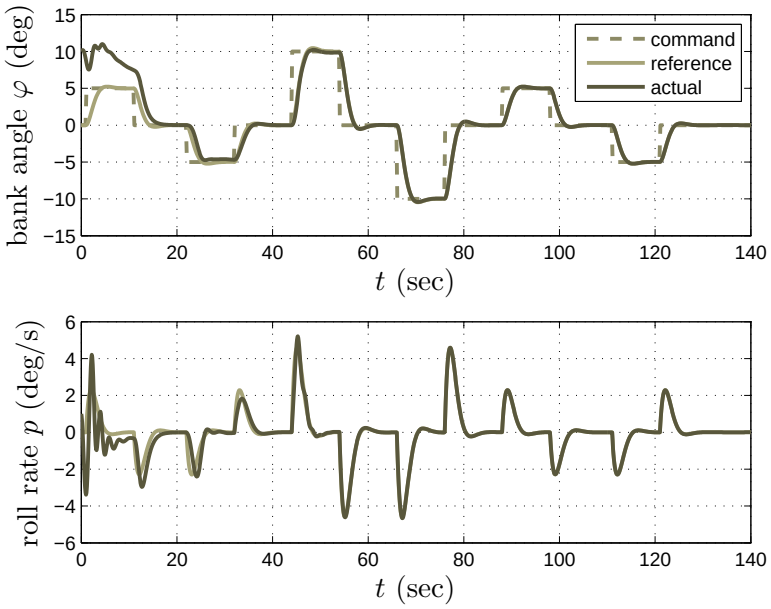


Figure 5.2: Adaptive controller on delta-wing dynamics

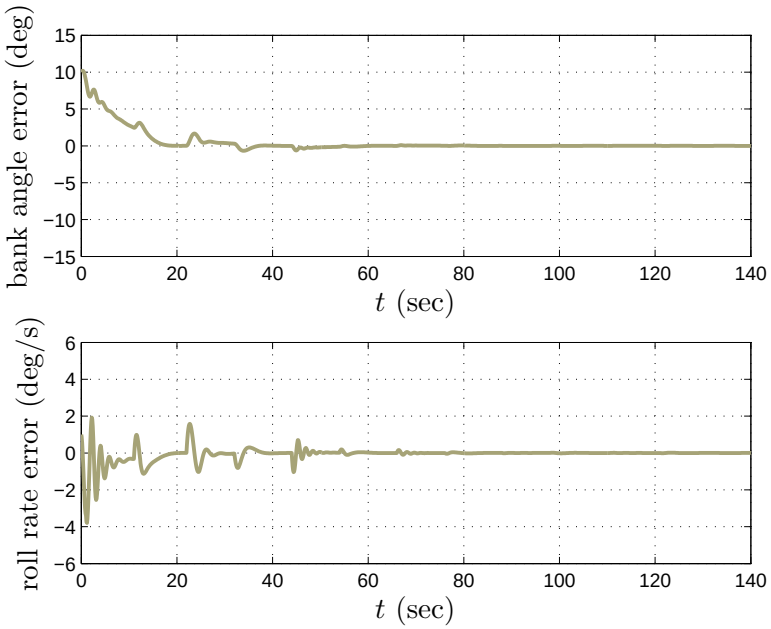


Figure 5.3: Error components $e = x - x^{\mathrm{ref}}$

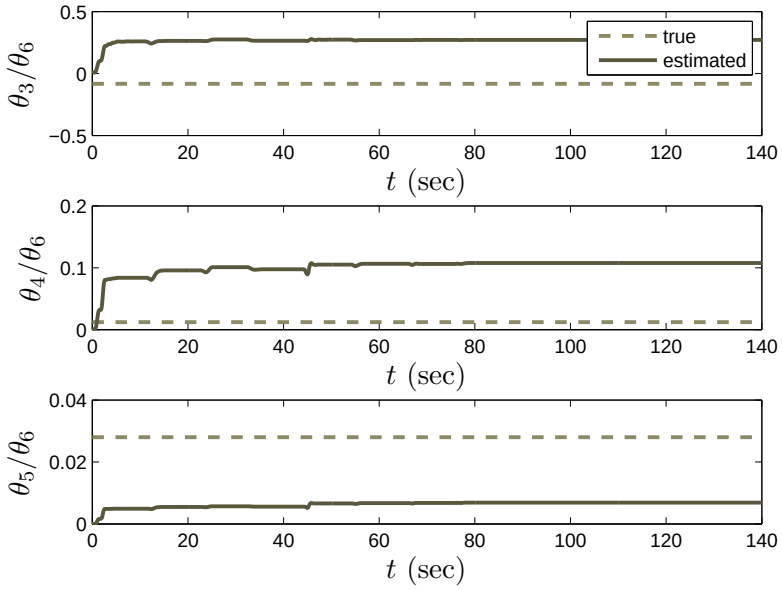


Figure 5.4: Parameter estimates

HJB inequalities for adaptive control, which parallel the lower bounds methods of previous chapters, appear in [WZL03], with applications to UAVs [Dyd10]. This chapter dealt primarily with SISO systems, however a great review of MRAC for MIMO systems is the paper [CHIK03]. In the machine learning literature, the algorithms in Section 5.3 are often called recursive least squares, stochastic gradient descent, and Kalman temporal difference learning.

Chapter 6

Networked Adaptive Systems

6.1 Introduction

We envision collaborative system identification applications where identical intelligent agents can communicate with each other, and are tasked with reaching consensus on some set of common parameters. While we are motivated by the case where these parameters specify the continuous dynamics for a nominal model class, of which every agent in the system is an instance, our information sharing framework readily applies to more general collaborative filtering and estimation schemes. For example, the parameters can refer to a static (or slowly changing) global state that each agent in the network can only partially observe. Either way, the parameters are determined in a decentralized, adaptive way through an online scheme that integrates local measurements with communicated information.

Classical (single agent) system identification algorithms determine model parameters by probing the system with an *a priori* selected input and observing the output. If the input is "exciting" enough to stimulate all the relevant internal dynamical modes, the model parameters can be backed out by adaptation. Otherwise, most algorithms can only ascertain the parameters to the extent that they replicate the observed input-output relationship. Designing an input that guarantees parameter convergence is difficult, because the persistence of excitation (PE) conditions that must be checked often require solving for the full system trajectory.

If we can replicate the system into an ensemble of identical systems and probe each one with a different input, can the parameter estimates converge to their true values under more relaxed conditions than with just one test system? In this chapter, we answer this question in the affirmative, provided that a collective persistence of excitation condition holds. The

condition ensures that a minimal "overall" level of input excitation is present within the communication network.

As a main ingredient of our collaborative identification scheme, we develop a parameter estimator based on a combination of linear consensus and local system identification. Consensus and flocking have been widely studied in computer science and dynamical systems, see [OSM04, OSFM07] and references therein for a good introduction. The closest works to ours, [SRS09, WS06], solve a sensor fusion problem in robotic coverage applications and characterize the role of persistently excited agents as knowledge leaders in the network. More recently, similar gradient-based schemes have been analyzed with noise in [SSS11] and sampled data in [GD13].

We expand upon classical parameter convergence results (*e.g.*, [MN77, And77, BS83, NA86, NA87]) by generalizing to the networked communication case. Our main contribution is a notion of *collective* persistence of excitation that takes advantage of the information shared between agents. As a special example of the condition, we show that parameter estimates can be made to converge to their true values even if no single agent uses a persistently exciting input.

6.1.1 Preliminaries

An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a finite set of n vertices $\mathcal{V} = \{v_1, \ldots, v_n\}$ together with a set of m edges $\mathcal{E} = \{e_1, \ldots, e_m\}$. We sometimes write i for v_i. An edge e_k is an unordered pair of vertices $\{v_i, v_j\} \subseteq \mathcal{V}$. The adjacency matrix of $\mathcal{G}$ is a matrix $A = [a_{ij}] \in \mathbf{R}^{n \times n}$ with entries

$$a_{ij} = \begin{cases} +1, & \text{if } \{v_i, v_j\} \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases}$$

The adjacency matrix is symmetric ($A = A^T$) for undirected graphs. The neighborhood of a vertex v_i consists of the set of adjacent vertex indices $\mathcal{N}_i = \{j \mid \{v_i, v_j\} \in \mathcal{E}\}$. The degree of v_i, written $\deg(v_i)$, is the number of neighbors $|\mathcal{N}_i|$ of that vertex, and the degree matrix is the diagonal matrix $D = \mathbf{diag}(\deg(v_1), \ldots, \deg(v_n)) \in \mathbf{R}^{n \times n}$.

The graph Laplacian $L = L^T \in \mathbf{R}^{n \times n}$ is defined as

$$L = D - A.$$

The Laplacian matrix is positive semidefinite, which we write as $L \succeq 0$, whenever $\mathcal{G}$ is connected. This follows from, *e.g.*, the fact that L is (weakly) diagonally dominant with strictly positive entries on the diagonal. In general, we write $A \succeq B$ to mean $A - B \succeq 0$ in the matrix sense.

A key property of connected graphs is that all eigenvalues $\lambda_1, \ldots, \lambda_n$ of L are strictly positive except for the smallest, which is zero. We order the eigenvalues of L as

$$0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n.$$

The second smallest eigenvalue λ_2 is the known as the algebraic connectivity of $\mathcal{G}$. The (column) eigenvector $\mathbf{1} = (1, \ldots, 1) \in \mathbf{R}^n$ corresponds to the zero eigenvalue subspace. In particular,

$$L\mathbf{1} = 0, \quad \mathbf{1}^T L = 0.$$

6.2 Problem setting

6.2.1 Parameter estimator dynamics

An ensemble of n agents has communication topology $\mathcal{G}$: each vertex v_i is an *agent* and each edge $e_k = \{v_i, v_j\}$ is an allowed (bidirectional) *communication link* between agents i and j. At any time $t \geq 0$, agent i can measure a surrogate state time series $x_i(t) \in \mathbf{R}^q$ and a real-valued output $y_i(t) \in \mathbf{R}$. The surrogate state $x_i(t)$ can be, for example, a filtered version of the agent's true dynamical state. We model the output $y_i(t)$ as a linear combination of parameters,

$$y_i(t) = \theta^T \phi(x_i(t)), \tag{6.1}$$

where $\phi : \mathbf{R}^q \to \mathbf{R}^p$ is a known regressor and $\theta \in \mathbf{R}^p$ is a vector of fixed but unknown coefficients.

In order to determine the parameter vector θ, each agent i has an estimate $\hat{\theta}_i(t)$ of θ made from local measurements and any information communicated by the agent's neighbors. The agent generates a local prediction of the output,

$$\hat{y}_i(t) = \hat{\theta}_i(t)^T \phi(x_i(t)), \tag{6.2}$$

and attempts to decrease its output prediction error $\tilde{y}_i = \hat{y}_i - y_i$ by modifying $\hat{\theta}_i$ with time. For brevity, define $\phi_i(t) = \phi(x_i(t))$. We consider the combined estimator dynamics

$$\begin{aligned}\frac{d}{dt}\hat{\theta}_i = &-\gamma\phi_i(t)(\hat{y}_i - y_i) \\ &+ \sum_{j\in\mathcal{N}_i} a_{ij}(\hat{\theta}_j - \hat{\theta}_i), \quad i = 1,\dots,n.\end{aligned} \tag{6.3}$$

With the first term of (6.3) we seek to reduce the local output prediction error. The constant estimation gain $\gamma > 0$ controls the local information fusion rate. Notice that this term is linear time-varying with the parameter estimates $\hat{\theta}_i$, which can be seen by substituting (6.2) into (6.3). The second term, a sum over the neighbors $\mathcal{N}_i$, presents a mechanism for global parameter consensus by ensuring that $\hat{\theta}_i$ does not stray too far from any neighboring $\hat{\theta}_j$, where $j \in \mathcal{N}_i$. In control theory terms, these dynamics describe a linear consensus controller driven by the learning signals $-\gamma\phi_i(t)(\hat{y}_i - y)$.

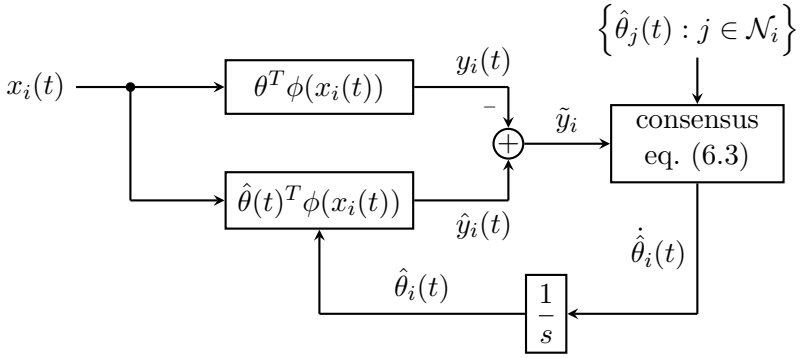


Figure 6.1: Each agent i in the network implements the estimator dynamics from eq. (6.3) and communicates only with its neighbors $\mathcal{N}_i$.

6.2.2 Persistence of excitation

The identified parameters governed by dynamics (6.3) asymptotically achieve consensus $\hat{\theta}_1 = \cdots = \hat{\theta}_n$ with all signals remaining bounded. In addition, if a collective *persistence of excitation* condition is met, the parameter errors $\tilde{\theta}_i = \hat{\theta}_i - \theta$ converge to zero. The convergence result is summarized in the following theorem.

Theorem 4. *Suppose that $\mathcal{G}$ is connected and each regressor $\phi_i(t) = \phi(x_i(t))$ remains bounded with bounded first derivative for all $i = 1,\dots,n$. Then the dynamics* (6.3) *exhibit*

1. bounded internal signals: $\hat{\theta}_i(t)$ *and* $\hat{y}_i(t)$ *are bounded for all* $i = 1, \ldots, n$ *and for all* $t \geq 0$,

2. asymptotic zero prediction error: $\tilde{y}_i(t) = \hat{y}_i(t) - y_i(t) \to 0$ *as* $t \to \infty$ *for all* $i = 1, \ldots, n$,

3. asymptotic parameter consensus: $\hat{\theta}_j(t) - \hat{\theta}_i(t) \to 0$ *as* $t \to \infty$ *for all* $i, j = 1, \ldots, n$.

If in addition there exist positive real numbers $m_1, m_2 > 0$ *such that for all* $t_0 \geq 0$ *and* $t > t_0$ *the matrix inequality*

$$m_2 I \succeq \frac{1}{t - t_0} \int_{t_0}^{t} \sum_{i=1}^{n} \phi_i(\tau)\phi_i(\tau)^T \, d\tau \succeq m_1 I \tag{6.4}$$

holds, then we also have

4. asymptotic parameter convergence: *the parameter errors* $\tilde{\theta}_i(t) = \hat{\theta}_i(t) - \theta \to 0$ *as* $t \to \infty$ *for all* $i = 1, \ldots, n$.

The hypothesis of Theorem 4 can also be rephrased with conditions on the regressor function itself, *e.g.*, $\phi : \mathbf{R}^q \to \mathbf{R}^p$ uniformly continuous in x.

Collective PE. The condition (6.4) encodes a notion of collective persistence of excitation (PE). For the trivial network with a single agent ($n = 1$), collective PE reduces to PE of a single regressor

$$m_2 I \succeq \frac{1}{t - t_0} \int_{t_0}^{t} \phi_1(\tau)\phi_1(\tau)^T \, d\tau \succeq m_1 I, \tag{6.5}$$

which is sufficient to obtain parameter convergence in traditional system identification where parameter consensus plays no explicit role [MN77, And77, BS83]. From linearity of the integral in condition (6.4), collective PE occurs in an ensemble $\{\phi_1, \ldots, \phi_n\}$ of regressors if, for example, any of the following types of excitation take place:

- *Enlightened*: a few ϕ_i are persistently exciting,
- *Total*: every ϕ_i is persistently exciting,
- *Intermittent*: there exists an unbounded sequence of times $t_1, t_2, \ldots$ such that some ϕ_i obeys (6.5) in each interval $[t_k, t_{k+1}]$,

- *Collaborative*: none of the ϕ_i is persistently exciting, but condition (6.4) still holds.

In the first two cases, distinguished agents have the role of a knowledge leader in the network (*cf.* [WS06]). The last two reveal that parameter convergence can still occur even if no single agent can claim leadership over all parameters, because the information shared through consensus reconciles any PE deficiency with other agents.

Example. Consider the sample communication network in Figure 6.2. Three agents are tasked with identifying a true parameter vector $\theta = (\theta_1, \theta_2) = (1, -1) \in \mathbf{R}^2$ using constant regressors. The system to be identified is $y_i(t) = \theta^T \phi_i(t)$. We let $\phi_i : [0, \infty) \to \mathbf{R}^2$ be given by $\phi_i(t) = (c_i, d_i)$, where c_i and d_i are fixed real constants for all $i = 1, 2, 3$.

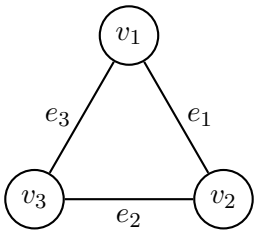


Figure 6.2: Communication graph with $n = 3$ agents and $m = 3$ links.

Each ϕ_i is not by itself persistently exciting, as the time average of a constant regressor outer product has rank one:

$$\frac{1}{t - t_0} \int_{t_0}^{t} \begin{bmatrix} c_i \\ d_i \end{bmatrix} \begin{bmatrix} c_i & d_i \end{bmatrix} d\tau = \begin{bmatrix} c_i \\ d_i \end{bmatrix} \begin{bmatrix} c_i & d_i \end{bmatrix} \not\succeq m_1 I$$

for any $m_1 > 0$; however, the collective PE condition (6.4) is still satisfied if the ϕ_i are not scalar multiples of the same vector,

$$m_2 I \succeq \frac{1}{t - t_0} \int_{t_0}^{t} \sum_{i=1}^{3} \begin{bmatrix} c_i \\ d_i \end{bmatrix} \begin{bmatrix} c_i & d_i \end{bmatrix} d\tau \succeq m_1 I$$

for some $m_1, m_2 > 0$. In other words, collective PE holds for constant regressors provided they span the parameter space $\mathbf{R}^2$. With rate $\gamma = 1$, the parameter estimates $\hat{\theta}_i \in \mathbf{R}^2$

evolve according to

$$\begin{cases} \dot{\hat{\theta}}_1 = -\phi_1(t)(\hat{y}_1 - y_1) + k(\hat{\theta}_2 - \hat{\theta}_1) + k(\hat{\theta}_3 - \hat{\theta}_1) \\ \dot{\hat{\theta}}_2 = -\phi_2(t)(\hat{y}_2 - y_2) + k(\hat{\theta}_3 - \hat{\theta}_2) + k(\hat{\theta}_1 - \hat{\theta}_2) \\ \dot{\hat{\theta}}_3 = -\phi_3(t)(\hat{y}_3 - y_3) + k(\hat{\theta}_1 - \hat{\theta}_3) + k(\hat{\theta}_2 - \hat{\theta}_3). \end{cases}$$

In the estimator dynamics above, consensus terms link the evolution of $\hat{\theta}_i$ to its neighboring $\hat{\theta}_j$ for $j \in \mathcal{N}_i$. Figure 6.3 illustrates the parameter estimates as a function of time for each of the three agents with ($k = 1$) and without ($k = 0$) consensus. We used the constant regressors

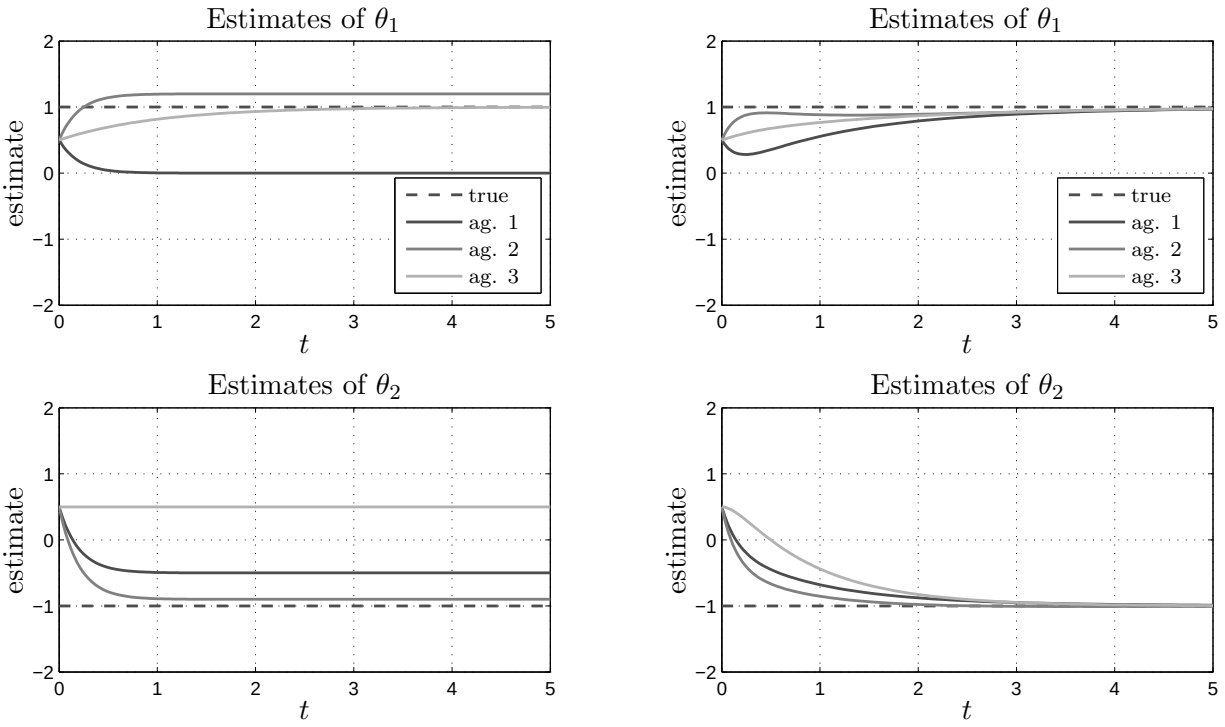
$$\phi_1(t) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \phi_2(t) = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \quad \phi_3(t) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Without consensus ($k = 0$, Figure 6.3(a)), individual parameter estimates depend solely on underdetermined measurements made at that node, so we have no reason to expect any $\hat{\theta}_i$ to converge to θ. With consensus ($k = 1$, Figure 6.3(b)), the agents collaboratively identify the true parameter.

Isolated agents develop their own (possibly inconsistent) parameter estimates, which replicate their observed input-output relationship. This is indicated in Figure 6.4 as a propensity toward the vertical axis. Parameter evolution is frozen once the output prediction error becomes zero, because the local prediction objectives J_i cannot be made any smaller. Collective PE and consensus allow both prediction error and parameter error to approach the origin by adding an extra regularization (disagreement) term to the objective.

Informal derivation. To see where the collective PE condition (6.4) originates, we sample the nominal system (6.1). Split the time interval $[t_0, t]$ into N equal intervals of length δ. For each agent $i = 1, \ldots, n$, we have the collection of N linear equations,

$$\begin{bmatrix} y_i(t_0) \\ y_i(t_0 + \delta) \\ y_i(t_0 + 2\delta) \\ \vdots \\ y_i(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \phi(x_i(t_0))^T \\ \phi(x_i(t_0 + \delta))^T \\ \phi(x_i(t_0 + 2\delta))^T \\ \vdots \\ \phi(x_i(t))^T \end{bmatrix}}_{\Psi_i} \begin{bmatrix} | \\ \theta \\ | \end{bmatrix}, \quad i = 1, \ldots, n. \tag{6.6}$$



(a) Without consensus

(b) With consensus

Figure 6.3: (a) Individual parameter estimates $\hat{\theta}_i$ fail to converge to θ in a network of three agents without a consensus mechanism in place, because each agent's input is not by itself persistently exciting. (b) With the same inputs and consensus, all parameter estimates to converge to the true value.

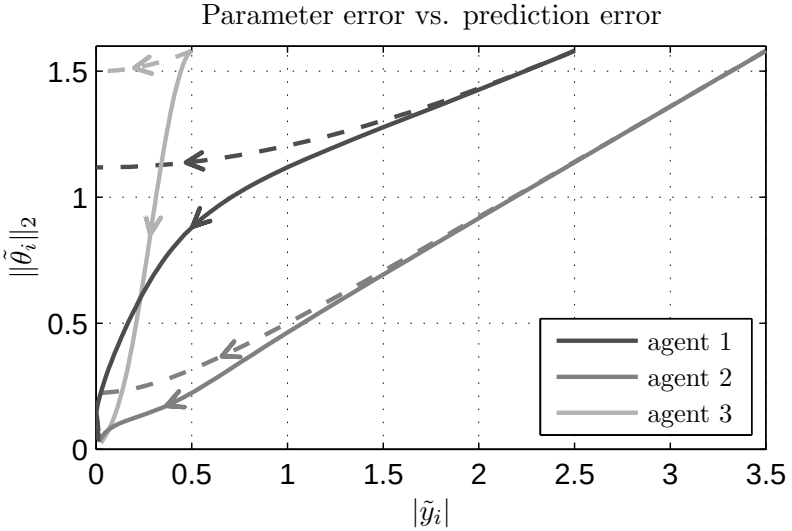


Figure 6.4: Prediction error (horizontal axis) tends to zero for all three agents with (solid) and without (dashed) consensus. Parameter error (vertical axis) also tends to zero with consensus due to collective PE.

The matrices $\Psi_i \in \mathbf{R}^{N\times p}$ in the linear system of equations (6.6) are skinny ($N \gg p$), so a unique least squares solution to (6.6) exists, provided the matrix

$$\Psi = \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \vdots \\ \Psi_n \end{bmatrix}$$

is full rank. Equivalently, the symmetric matrix $\Psi^T\Psi$ should be positive definite. In terms of the sampled regressors, we require

$$\sum_{k=0}^{N-1}\sum_{i=1}^{n} \phi(x_i(t+k\delta))\phi(x_i(t+k\delta))^T \succeq N\alpha I$$

for some real constant $\alpha > 0$. Multiplying both sides by δ and taking $N \to \infty$, $\delta \to 0$ gives the integral condition (6.4), as required.

Instantaneous objective minimization. We reinterpret the dynamics (6.3) as instantaneous minimization of a particular cost function. The dynamics arise from two main desires for the network as a whole. First, all local estimates $\hat{\theta}_i$ should converge to the same value as $t \to \infty$, and second, the value to which the local estimates converge should be the true θ. Define at each time $t \geq 0$ an instantaneous quadratic cost $J : \mathbf{R}^p \times \cdots \times \mathbf{R}^p \to \mathbf{R}$,

$$\begin{aligned} &J(\hat{\theta}_1(t), \ldots, \hat{\theta}_n(t)) = \\ &\qquad \sum_{i=1}^{n} \gamma J_i(\hat{\theta}_i(t)) + \sum_{\{v_i,v_j\}\in\mathcal{E}} \frac{a_{ij}}{2}\left\|\hat{\theta}_j(t) - \hat{\theta}_i(t)\right\|_2^2, \end{aligned} \tag{6.7}$$

with variables $\hat{\theta}_i(t) \in \mathbf{R}^p$ and local prediction costs $J_i : \mathbf{R}^p \to \mathbf{R}$ for all $i = 1, \ldots, n$. The dynamics (6.3) can be recovered from the gradient flow

$$\frac{d}{dt}\hat{\theta}_i = -\frac{\partial J}{\partial \hat{\theta}_i}, \quad i = 1, \ldots, n, \tag{6.8}$$

with quadratic local prediction costs

$$J_i(\hat{\theta}_i(t)) \triangleq \frac{1}{2}\big(\hat{y}_i(t) - y_i(t)\big)^2, \quad i = 1, \ldots, n,$$

where $\hat{y}_i(t)$ is given in terms of $\hat{\theta}_i(t)$ by the local prediction equation (6.2), and $y_i(t)$ comes from an online measurement. The learning rate $\gamma > 0$ trades off instantaneous prediction error with total parameter disagreement.

6.2.3 Proof of Theorem 4 for $p = 1$

To show the main ideas, and following [PLM14], we will prove Theorem 4 for the case θ is a scalar ($p = 1$). The vector case ($p > 1$) is not substantially different, and is included in Appendix A for completeness. The proof relies on a standard result of analysis that the derivative of a function that has a finite limit converges to zero, if that derivative is also uniformly continuous, *e.g.*, [KHA02]. Known as Barbalat's lemma, restated below in its integral form, it is central to proving Theorem 4, parts 1–3.

Lemma (Barbalat)**.** *Let $f : [0, \infty) \to \mathbf{R}$ be a uniformly continuous function and suppose that $\lim_{t\to\infty} \int_0^t f(\tau)\, d\tau$ exists and is finite. Then $f(t) \to 0$ as $t \to \infty$.*

To obtain parameter convergence in Theorem 4, part 4, we will use the persistence of excitation condition from [MN77], which gives necessary and sufficient conditions for the uniform asymptotic stability of a time-varying autonomous system. It says that the origin is the unique stable equilibrium of $\dot{x} = -P(t)x$, if the matrix $-P(t) \in \mathbf{R}^{n\times n}$ is stable, on average, in any direction in $\mathbf{R}^n$. The condition below can be expressed in many ways, and we direct the reader to the classical references [MN77, AND77, BS83, SB89] for additional insight.

Theorem 5 (Morgan and Narendra 1977)**.** *Suppose $P(t)$ is a symmetric positive semidefinite matrix of bounded piecewise continuous functions. Then the equation $\dot{x} = -P(t)x$ is uniformly asymptotically stable if and only if there are real numbers $a > 0$ and b such that for all $t_0 \geq 0$ and $t \geq t_0$,*

$$\int_{t_0}^{t} w^T P(\tau) w \, d\tau \geq a(t - t_0) + b$$

for all fixed unit vectors w.

Proof of Theorem 4 for $p = 1$. Stack the real components $\hat{\theta}_i$ into a column vector $\hat{\theta} = (\hat{\theta}_1, \ldots, \hat{\theta}_n) \in \mathbf{R}^n$ and let the parameter error be $\tilde{\theta} = \hat{\theta} - \theta\mathbf{1} \in \mathbf{R}^n$. In view of the estimator definitions in (6.2), each agent's individual learning signal depends on $\tilde{\theta}$ via

$$-\gamma\phi(x_i)(\hat{y}_i - y_i) = -\gamma\phi(x_i)^2\tilde{\theta}_i.$$

Putting these together, the individual dynamics (6.3) can be aggregated in matrix form as

$$\frac{d}{dt}\tilde{\theta} = -L\tilde{\theta} - \gamma\Phi\tilde{\theta}, \tag{6.9}$$

where $\Phi(t) = \mathbf{diag}(\phi_1^2(t), \ldots, \phi_n^2(t)) \in \mathbf{R}^{n\times n}$, and we used the identity $L\tilde{\theta} = L(\hat{\theta} - \theta\mathbf{1}) = L\hat{\theta}$.

Consider the candidate Lyapunov function

$$V(\tilde{\theta}) = \frac{1}{2}\tilde{\theta}^T\tilde{\theta}.$$

The time derivative of V along solution trajectories of (6.9) is

$$\begin{aligned}
\dot{V}(\tilde{\theta}) &= \frac{1}{2}\left(\left(\frac{d}{dt}\tilde{\theta}\right)^T\tilde{\theta} + \tilde{\theta}^T\left(\frac{d}{dt}\tilde{\theta}\right)\right) \\
&= \frac{1}{2}\Big(\big(-L\tilde{\theta} - \gamma\Phi\tilde{\theta}\big)^T\tilde{\theta} \\
&\qquad + \tilde{\theta}^T\big(-L\tilde{\theta} - \gamma\Phi\tilde{\theta}\big)\Big) \\
&= -\tilde{\theta}^T L\tilde{\theta} - \gamma\tilde{\theta}^T\Phi\tilde{\theta} \\
&\leq 0,
\end{aligned} \tag{6.10}$$

where the inequality follows from the positive semidefiniteness of L and $\Phi(t)$ and from the learning rate assumption $\gamma > 0$.

Since V is bounded below ($V \geq 0$) and nonincreasing ($\dot{V} \leq 0$), it converges to a limit as $t \to \infty$. Furthermore, V is uniformly bounded above by its initial value, because

$$\begin{aligned}
V\big(\tilde{\theta}(t)\big) &= V\big(\tilde{\theta}(0)\big) + \int_0^t \underbrace{\dot{V}\big(\tilde{\theta}(\tau)\big)}_{\leq 0}\, d\tau \\
&\leq V\big(\tilde{\theta}(0)\big),
\end{aligned}$$

hence $\tilde{\theta}$ is bounded, from which we conclude that the local estimates $\hat{\theta}_i = \tilde{\theta}_i + \theta$ and predictions $\hat{y}_i = \hat{\theta}_i\phi(x_i)$ are bounded. This finishes the proof of part 1. Next, we integrate both sides of (6.10),

$$V(t) - V(0) = -\int_0^t \tilde{\theta}(\tau)^T L\tilde{\theta}(\tau) + \gamma\sum_{i=1}^n |\tilde{y}_i(\tau)|^2\, d\tau,$$

and let $t \to \infty$. Note that the prediction errors $\tilde{y}_i(t)$ are square integrable for all $i = 1, \ldots, n$. Moreover, the quadratic disagreement $\tilde{\theta}^T L \tilde{\theta}$ has a finite integral. If we can prove uniform continuity of $\tilde{y}_i$ and $\tilde{\theta}^T L \tilde{\theta}$, then Barbalat's lemma would imply parts 2 and 3.

The derivative of $\tilde{y}_i$ is

$$\frac{d}{dt}\tilde{y}_i = \Big(\frac{d}{dt}\tilde{\theta}_i\Big)^T \phi(x_i(t)) + \tilde{\theta}_i^T D\phi(x_i(t))\dot{x}_i(t), \tag{6.11}$$

where $D\phi(x_i(t)) \in \mathbf{R}^{p\times q}$ is the Jacobian matrix of ϕ with respect to x evaluated at $x_i(t)$. Since $d\tilde{\theta}/dt$ is bounded as a result of (6.9), and $x_i(t)$, $\dot{x}_i(t)$ are bounded by assumption with ϕ continuously differentiable with respect to x, the derivative (6.11) is bounded. Thus $\tilde{y}_i \to 0$ as $t \to \infty$, proving part 2. Next,

$$\frac{d}{dt}\tilde{\theta}^T L \tilde{\theta} = -\tilde{\theta}^T (2L^T L + \gamma(\Phi L + L\Phi))\tilde{\theta} \tag{6.12}$$

is bounded because it is a sum of bounded terms, thus $\tilde{\theta}^T L \tilde{\theta} \to 0$ as $t \to \infty$. In particular, this means $L\tilde{\theta} = L\hat{\theta} \to 0$, where again we used $L\mathbf{1} = 0$. For a connected graph, the null space of the Laplacian is $\text{null}(L) = \text{span}\{\mathbf{1}\}$, hence $\hat{\theta}_j - \hat{\theta}_i \to 0$ for all $i, j = 1, \ldots, n$. In other words, the parameter estimates asymptotically reach consensus. This completes the proof of part 3.

For part 4, note that the dynamics of $\tilde{\theta}$ in (6.9) are linear time-varying, so it suffices to show that the condition in Theorem 5 is met for $P(t) = L + \gamma\Phi(t)$ and $b = 0$.

Let the Laplacian have eigendecomposition $Lv_i = \lambda_i v_i$ with $\lambda_i > 0$ for $i = 2, \ldots, n$. Complete the basis of $\mathbf{R}^n$ so $\{\frac{1}{\sqrt{n}}\mathbf{1}, v_2, \ldots, v_n\}$ is an orthonormal set. Write a unit vector w in this basis as

$$w = \frac{\alpha}{\sqrt{n}}\mathbf{1} + \sum_{j=2}^{n} \beta_j v_j, \tag{6.13}$$

so that $(\alpha, \beta_2, \ldots, \beta_n) \in \mathbf{R}^n$ has unit norm. Pick $t_0 \geq 0$ and $t > t_0$, and denote the time average of a quantity over the interval $[t_0, t]$ by a bar over the quantity, as in

$$\bar{\Phi} \triangleq \frac{1}{t - t_0}\int_{t_0}^{t} \Phi(\tau)\, d\tau,$$

so that collective PE (6.4) implies $m_2 \geq \mathbf{1}^T\bar{\Phi}\mathbf{1} \geq m_1$. Then,

$$\frac{1}{t-t_0}\int_{t_0}^{t} w^T(L+\gamma\Phi(\tau))w\,d\tau$$
$$= w^TLw + w^T\bar{\Phi}w \geq \max\{w^TLw, \gamma w^T\bar{\Phi}w\}, \tag{6.14}$$

because L and $\bar{\Phi}$ are positive semidefinite. Our goal is to bound the maximum (6.14) away from zero for all time. By substituting (6.13) into (6.14) and using $L\mathbf{1} = 0$ and $\mathbf{1}^TL = 0$, we bound the first term in the maximum below by

$$\begin{aligned} w^TLw &= \sum_{i=2}^{n}\beta_i v_i^TL\sum_{j=2}^{n}\beta_j v_j \\ &= \sum_{i=2}^{n}\sum_{j=2}^{n}\beta_i\beta_j\lambda_j v_i^Tv_j = \sum_{i=2}^{n}\lambda_i\beta_i^2 \\ &\geq \lambda_2\|\beta\|_2^2 \\ &= \lambda_2(1-\alpha^2), \end{aligned}$$

where $\beta = (\beta_2, \ldots, \beta_n) \in \mathbf{R}^{n-1}$, and in the last line we used $\|\beta\|_2^2 = 1-\alpha^2$. Next, for $V \stackrel{\Delta}{=} [v_2, \ldots, v_n] \in \mathbf{R}^{n\times n-1}$, the second term has a lower bound

$$\begin{aligned} w^T\bar{\Phi}w &= \frac{\alpha^2}{n}\mathbf{1}^T\bar{\Phi}\mathbf{1} + \underbrace{\beta^TV^T\bar{\Phi}V\beta}_{\geq 0} + \frac{2\alpha}{\sqrt{n}}\mathbf{1}^T\bar{\Phi}V\beta \\ &\geq \frac{\alpha^2}{n}\underbrace{\mathbf{1}^T\bar{\Phi}\mathbf{1}}_{\geq m_1} - \frac{2|\alpha|}{\sqrt{n}}|\mathbf{1}^T\bar{\Phi}V\beta| \\ &\geq \frac{\alpha^2}{n}m_1 - \frac{2|\alpha|}{\sqrt{n}}\underbrace{\|\bar{\Phi}\mathbf{1}\|_1}_{\leq m_2}\underbrace{\|V\beta\|_\infty}_{\leq\|\beta\|_2} \\ &\geq \frac{\alpha^2}{n}m_1 - 2m_2\sqrt{\frac{\alpha^2}{n}(1-\alpha^2)}. \end{aligned}$$

The second line follows form Cauchy-Schwarz and the third from Hölder's inequality. Putting these together gives the required lower bound

$$\max\{w^TLw, \gamma w^T\bar{\Phi}w\} \geq a > 0,$$

where the worst case rate constant a is

$$a = \inf_{|\alpha|\leq 1} \max\Big\{\lambda_2(1-\alpha^2), \\ \gamma\frac{\alpha^2}{n}m_1 - 2\gamma m_2\sqrt{\frac{\alpha^2}{n}(1-\alpha^2)}\Big\}. \tag{6.15}$$

Note that the infimum in (6.15) is attained by continuity, and is strictly positive (if a and the first term is zero, then so is the second, $\gamma m_1/n = 0$, a contradiction). □

6.3 Interpretations and related problems

We present quadratic J_i for simplicity, though it is straightforward to devise schemes robust to process noise, communication noise, unmodeled dynamics, and parameter drift, see [IF06, SB89, LW13]. For example, if instead we use

$$J_i(\hat{\theta}_i(t)) \triangleq \int_0^t \left(\hat{\theta}_i(t)^T\phi(x_i(\tau)) - y_i(\tau)\right)^2 d\tau$$

in the cost (6.7), our collaborative identification scheme becomes more robust to measurement noise. If the prediction costs are zero ($J_i = 0$) and $p = 1$, the flow (6.8) reduces to gradient flow on the quadratic disagreement function

$$J(\hat{\theta}_1, \dots, \hat{\theta}_n) = \sum_{\{v_i,v_j\}\in\mathcal{E}} \frac{a_{ij}}{2}\left(\hat{\theta}_j - \hat{\theta}_i\right)^2,$$

which is the classical linear consensus flow $\dot{\hat{\theta}} = -L\hat{\theta}$, see [OSM03, OSM04, OSFM07]. We can also reformulate the cost (6.7) as a constrained objective, rather than a quadratically penalized objective, to obtain second order dynamics (PI control) with parameter consensus [YCCK10, Ren08].

Augmented Lagrangian flow. Consider at each each time t the optimization problem

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^n J_i(\hat{\theta}_i(t)) \\ &\text{subject to} && \hat{\theta}_j(t) - \hat{\theta}_i(t) = 0, \quad \{v_i, v_j\} \in \mathcal{E} \end{aligned} \tag{6.16}$$

with variables $\hat{\theta}_i(t) \in \mathbf{R}^p$ and local identification objectives $J_i : \mathbf{R}^p \to \mathbf{R}$ for all $i = 1, \ldots, n$. The quadratic cost (6.7) can be recast as an augmented Lagrangian of the optimization (6.16),

$$\begin{aligned} \mathcal{L}(\hat{\theta}, \nu) = &\sum_{i=1}^{n} J_i(\hat{\theta}_i) \\ &+ \sum_{\{v_i, v_j\} \in \mathcal{E}} \Big(\nu_k^T (\hat{\theta}_j - \hat{\theta}_i) + \frac{a_{ij}}{2} \|\hat{\theta}_j - \hat{\theta}_i\|_2^2 \Big), \end{aligned} \tag{6.17}$$

where the index k corresponds uniquely to edge $e_k = \{v_i, v_j\}$ and the multiplier ν_k.

Solving (6.16) amounts to finding a saddle point of the augmented Lagrangian (6.17), which can be attained via the min-max flow

$$\begin{cases} \dot{\hat{\theta}}_i = -\dfrac{\partial \mathcal{L}}{\partial \hat{\theta}_i}, & i = 1, \ldots, n \\ \dot{\nu}_k = +\dfrac{\partial \mathcal{L}}{\partial \nu_k}, & k = 1, \ldots, m. \end{cases} \tag{6.18}$$

We see immediately the utility of the quadratic augmentation term: it gives rise to the proportional neighbor terms in the individual dynamics (6.3). Each dual variable gives rise to one of m edge constraints, one for communication link. This can be interpreted as PI parameter control scheme [Cor08].

Extension to dynamical systems. The persistence of excitation condition (6.4) is awkward to verify for generic dynamical systems even in the classical single agent setting ($n = 1$). For linear dynamical systems, it can be shown that a sinusoidal input with enough independent frequency components, *i.e.*, an input that is *sufficiently rich*, will generate the persistence of excitation necessary for parameter convergence [SB89, YW77].

We now demonstrate how sufficient richness translates to the collaborative multi agent setting by example. Suppose the nominal system to be identified is the (stable) single input linear system

$$\dot{x}(t) = ax(t) + bu(t),$$

where $a < 0$ and b are constant (unknown) parameters. With our goal to determine a and b, we instantiate an ensemble of identical systems whose communication structure is organized by the graph $\mathcal{G}$. Each agent chooses their own input $u_i(t) \in \mathbf{R}$ and observes the

resulting state $x_i(t) \in \mathbf{R}$ for all $i = 1, \ldots, n$. The dynamics are

$$\dot{x}_i(t) = ax_i(t) + bu_i(t), \quad i = 1, \ldots, n. \tag{6.19}$$

Note that the dynamics (6.19) are of the nominal form (6.1), where $y_i(t) \in \mathbf{R}$ is the state derivative $\dot{x}_i(t)$, the regressor is $\phi_i(t) = (x_i(t), u_i(t)) \in \mathbf{R}^2$, and the unknown parameter vector is $\theta = (a, b) \in \mathbf{R}^2$. In practice, the time derivative $\dot{x}_i(t)$ is not a signal available for measurement, so we often (linearly) filter both sides of (6.19) and redefine surrogate outputs and regressors by their filtered versions.

In the classical setting ($n = 1$), condition (6.4) is satisfied if we choose $u(t) = \sin(\omega t)$ with $\omega \neq 0$, because

$$\frac{1}{t - t_0} \int_{t_0}^{t} \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} \begin{bmatrix} x(\tau) & u(\tau) \end{bmatrix} d\tau$$

eventually has bounded positive eigenvalues. Note that the choice of $u(t)$ determines $x(t)$, and hence the value of the integral above. In the multi agent setting ($n > 1$), there is considerably more design freedom in choosing the inputs $u_i(t)$ to obtain desired parameter convergence dynamics while maintaining collective PE, and hence a guarantee of parameter convergence.

For example, it suffices that $u_i(t) = \sin(\omega t)$ for some $i \in \{1, \ldots, n\}$, while the rest of the $u_j(t)$, for $j \neq i$, are arbitrary. The distinguished agent i can be thought of as *enlightened* to the true dynamics of the system because that agent is probed with a known sufficiently rich input. Parameter consensus then ensures that all other agents reach the same conclusion about the values of a and b as the enlightened agent i. Moreover, if all agents are enlightened, as is the case in *total* excitation, the designer of the collaborative identification system can trade off parameter dynamics (time) against the number of agents (space).

Deadzone. Proposed in [PN82] and commonly used in practical adaptive systems, a deadzone modification is readily incorporated into the dynamics (6.3),

$$\frac{d}{dt}\hat{\theta}_i = \begin{cases} -\gamma\phi_i(t)(\hat{y}_i - y_i) + \displaystyle\sum_{j \in \mathcal{N}_i} a_{ij}(\hat{\theta}_j - \hat{\theta}_i), & |\hat{y}_i - y_i| \geq e_{\min} \\ 0 & |\hat{y}_i - y_i| < e_{\min}, \end{cases}$$

where $e_{\min}$ is some minimum level of output tracking error. This simple, but important modification ensures that $\tilde{\theta}_i$ does not run away in magnitude when the tracking error is small.

Different robustness properties are emphasized if a deadzone is placed on the learning or network term, individually. For example, if a deadzone is placed on the network term, this allows for θ to be slightly different agent to agent.

σ-mod. The network consensus terms provide a level of damping and robustness against noise. Extra damping, however, can sometimes be beneficial [IK83],

$$\frac{d}{dt}\hat{\theta}_i = -\gamma\phi_i(t)(\hat{y}_i - y_i) + \sum_{j\in\mathcal{N}_i} a_{ij}(\hat{\theta}_j - \hat{\theta}_i) - \sigma\hat{\theta}_i,$$

where $\sigma > 0$ is a constant damping parameter.

e-mod. A nonlinear damping scaled by the output reconstruction error prevents the tendency for $\hat{\theta}_i$ to go to zero when the reconstruction error is small [NA87],

$$\frac{d}{dt}\hat{\theta}_i = -\gamma\phi_i(t)(\hat{y}_i - y_i) + \sum_{j\in\mathcal{N}_i} a_{ij}(\hat{\theta}_j - \hat{\theta}_i) - \sigma|\hat{y}_i - y|\hat{\theta}_i,$$

where $\sigma > 0$ is a constant damping parameter.

Regressor selection. Well chosen regressors, corresponding to well chosen inputs, lead to accelerated identification by spreading the work of system identification across multiple agents, while poorly chosen inputs can result in the network "fighting" against individuals. Experimental and theoretical investigation of the interplay among agents can shed a more nuanced light on the benefits as well as limitations of the proposed collaborative identification scheme.

As an example of the types of phenomena observed in our studies, note that in the proof of Theorem 4 we showed that the worst-case convergence rate constant (6.15) was strictly positive. It is known that the bound is not tight. Consider the problem of choosing optimal

constant regressors to maximize the convergence rate, obtained by solving the optimization

$$\begin{aligned} \text{maximize} \quad & \lambda_{\min}(L \otimes I_p + \gamma\Phi) \\ \text{subject to} \quad & \Phi = \begin{bmatrix} \phi_1\phi_1^T & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi_n\phi_n^T \end{bmatrix} \\ & 0 \preceq \Phi \preceq I_{np}, \end{aligned} \tag{6.20}$$

with variables $\Phi \in \mathbf{R}^{np\times np}$ and $\phi_1, \ldots, \phi_n \in \mathbf{R}^p$, where $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of a matrix and $\otimes$ is the Kronecker product. The optimal values $\phi_1^\star, \ldots, \phi_n^\star$ of (6.20) correspond to a choice of constant regressors for the n agents that maximizes the convergence rate of the estimator dynamics (6.3).

The rank requirements embedded within the first (sparsity pattern) constraint make the regressor selection problem nonconvex. Using an ADMM heuristic [BPC+11] for the specific three-agent example in §6.2.2, we come up with a better regressor choice,

$$\phi_1^\star(t) = \begin{bmatrix} -0.63037 \\ -0.77629 \end{bmatrix}, \quad \phi_2^\star(t) = \begin{bmatrix} 0.47497 \\ -0.88 \end{bmatrix}, \quad \phi_3^\star(t) = \begin{bmatrix} -0.9894 \\ 0.14525 \end{bmatrix}.$$

In Figure 6.5, we plot the rate bound (6.15) against the parameter α, zoomed to $\alpha \in [0.75, 1]$ to show detail, and compare it to the rate bound from a heuristic regressor selection scheme that approximately solves optimization (6.20).

Note that $\phi_1^\star, \phi_2^\star, \phi_3^\star$ achieve faster convergence than ϕ_1, ϕ_2, ϕ_3, but both regressor sets admit a choice of α for which the convergence rate of the dynamics (6.3) is poor. Examine once again the bound (6.15). At $\alpha = 0$, the convergence rate is the algebraic connectivity λ_2 of the communication graph $\mathcal{G}$, while at $\alpha = 1$, the convergence rate is

$$a = \gamma\frac{m_1}{n},$$

a function of the minimum excitation level m_1 and the learning rate γ. In Fig 6.5, we suggest that there exist directions in parameter space ($\alpha \approx 0.957$) where consensus and adaptation disastrously conspire against each other, whereas other directions give better convergence properties.

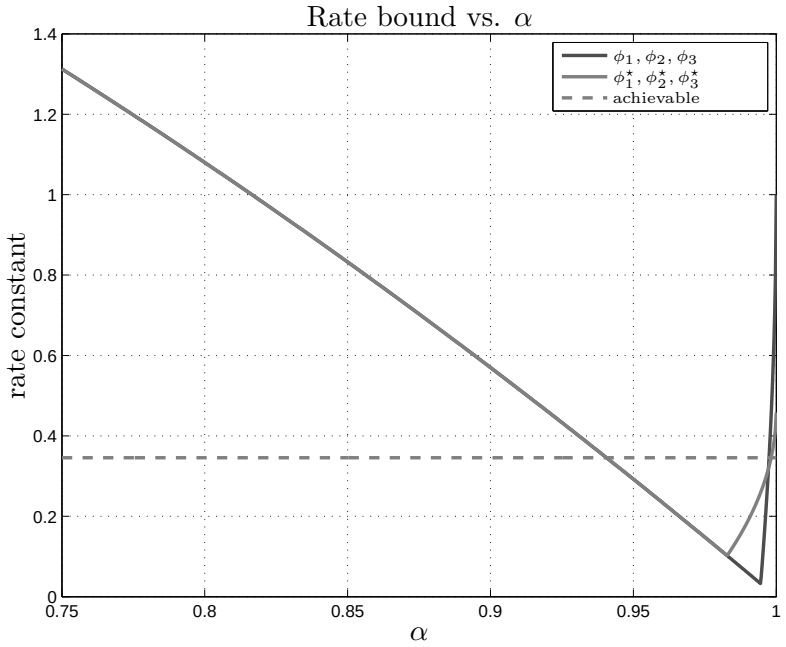


Figure 6.5: Rate bound (solid) for two different regressor choices, and the actual convergence rate (dashed) with regressor choice $\phi_1^\star, \phi_2^\star, \phi_3^\star$.

Optimal input to achieve PE. In a time-limited identification scenario, such as in fault detection, it is necessary to identify the system as quickly as possible so appropriate control actions can be taken to survive the fault. In these cases, the excitation input can be chosen to maximize the parameter convergence rate.

More concretely, consider the discrete time, single agent, version of our problem, with a linear in parameters system model

$$y_k = \theta^T \phi(x_k), \quad k = 0, 1, \ldots,$$

and the estimation process

$$\begin{aligned} \hat{y}_k &= \hat{\theta}_k^T \phi(x_k), \\ \hat{\theta}_{k+1} &= \hat{\theta}_k - \Gamma \phi(x_k) \left(\hat{y}_k - y_k \right), \quad t = 0, 1, \ldots. \end{aligned}$$

We would like to estimate the fixed parameter vector $\theta \in \mathbf{R}^p$, where $x_k = (\cdot, u_k) \in \mathbf{R}^q$ is the surrogate state, y_k is a real output, $\Gamma = \Gamma^T \succ 0$ is a given matrix, and the regressor function $\phi : \mathbf{R}^q \to \mathbf{R}^p$ is uniformly continuous in x_k.

The parameter error $\tilde{\theta}_k = \hat{\theta}_k - \theta$ obeys the recurrence

$$\tilde{\theta}_{k+1} = (I - \Gamma\phi(x_k)\phi(x_k)^T)\tilde{\theta}_t.$$

We have a PE-like convergence result [LN88]: if there exists an integer interval width $L > 0$ and a constant $\alpha > 0$ such that

$$\frac{1}{L}\sum_{k=j}^{j+L-1} \phi(x_k)\phi(x_k)^T \succeq \alpha I, \quad \text{for all } j = 1, 2, \ldots$$

then the parameter error $\tilde{\theta}_k$ converges to zero as $k \to \infty$. The per-step convergence factor

$$\rho(M_k) = \rho(I - \Gamma\phi(x_k)\phi(x_k)^T)$$

where $\rho(\cdot)$ denotes the spectral radius of a matrix, is a proxy for the rate at which the parameter error tends to zero.

Consider at each time k the per-step optimization problem

$$\begin{array}{ll} \text{minimize} & \rho(M_k) = \rho(I - \Gamma\phi(x_k)\phi(x_k)^T) \\ \text{subject to} & u_k \in \mathcal{C} \end{array} \tag{6.21}$$

which greedily picks an input $u_k \in \mathcal{C}$ at each step in order to maximize the convergence rate. Interestingly although this formulation does not explicitly mention networks, similar problems have been widely studied in the networked and robust control literature [XB04, XBL05, DS81, PD93, DP00].

Using the identity $\rho(M_k) \leq \gamma$ if and only if $-\gamma I \preceq M_k \preceq \gamma I$, we can rewrite problem (6.21) as

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & -\gamma I \preceq I - \Gamma\phi(x_k)\phi(x_k)^T \preceq \gamma I \\ & u_k \in \mathcal{C}. \end{array}$$

Multiplying all sides of the first constraint by Γ^{-1} and taking a Schur complement, we

obtain the equivalent bilinear SDP

$$\begin{aligned} \text{minimize} \quad & \gamma \\ \text{subject to} \quad & \begin{bmatrix} I & \phi(x_k)^T \\ \phi(x_k) & (1+\gamma)\Gamma^{-1} \end{bmatrix} \succeq 0 \\ & \Gamma^{-1} - \phi(x_k)\phi(x_k)^T \preceq \gamma\Gamma^{-1} \\ & u_k \in \mathcal{C}. \end{aligned}$$

The bilinear constraint $\Gamma^{-1} - \phi(x_k)\phi(x_k)^T \preceq \gamma\Gamma^{-1}$ is nonconvex and thus difficult to enforce in practice. For small dimensions, branch and bound, as well as powerful multiplier methods are readily applied to solve the problem [NW06, Ber05, Ber76, Ber96, BPC+11].

Chapter 7

Conclusion

7.1 Summary and contributions

This thesis was informed by the realization that the frontiers of control systems are no longer in writing down a formula or a PDE, solving it once, and implementing it on the system—but rather in designing a sequence of steps, a method, an *algorithm* to solve the control problem in the loop, along with a proof that the algorithm will give a correct or nearly correct answer. As computers get faster and more pervasive, online optimization for control is coming to systems that were traditionally not part of the control designer's responsibility. Whether informed by careful a priori modeling, adaptation, or by gathered data, engineers need the right language to specify and provide for closed loop stability. This thesis argues that the right language of Lyapunov functions has been known for over a hundred years, with the minor caveat that to make the language actionable, the great utility of bounds and approximate policies cannot be overlooked.

Specifically, upper bounds on the Lyapunov function lead to policies that are *robust* by construction. If an upper bound cannot be found, then either the closed loop system is not robust with respect to the provided parameterization, requiring a re-thinking of the robustness model, or the system is so fragile that it cannot be stabilized. On the other hand, lower bounds result in inherently infeasible performance bound policies, which have no guarantees of stability or performance, however they lead to approximate policies that in practice allow for sophisticated *adaptation* and *learning*. This thesis showed novel applications of lower bounds to temporal logic constrained systems, and upper bounds to adaptive systems, providing approximate solutions without resorting to intractable discretizations. It was shown that constraints in adaptive systems lead to improved notions of identifiability,

meanwhile networks themselves can be a source of extra performance and robustness. The story and applications are very exciting, and we are barely touching their surface.

7.2 On the title

Although it is already happening to some extent, it is my sincere wish that some day control theory, machine learning, and optimization will all use the same language to describe a greater theory of computational mathematics of which they are all a small part. The title of this thesis reflects the key words that I can reasonably see coalescing in mutually beneficial harmony.

Why "robustness"? Robust control is a well-developed theory that any student of control must know in order to implement control systems safely. Closed loop stability is meaningless in practice without some sort of quantifiable margins. This thesis argues that robust control is the study of Lyapunov upper bounds, specifically ones obtained offline.

Why "adaptation"? Control theorists know a lot about dynamical systems from Lyapunov and dissipation bounds, however for the most part they do not yet know the right way to treat incoming data. Thus adaptive control still has much to learn from machine learning and optimization.

Why "learning"? Machine learning professionals know a lot about the right way to treat incoming data, however they are often less worried about closed loop stability and systems concepts than control theorists. Moreover, just as the goal of "learning" is to synthesize concepts and make them simpler, so does choosing simple Lyapunov function candidates result in simple approximate policies. Adaptation and learning are two aspects of the same not yet well described underlying theory, with complementary knowledge bases yet to be shared.

Why "in optimal control"? Optimization, specifically convex optimization, is becoming less a tool in control, and more of a language of control. Variationally speaking, many traditional concepts like controllability and $\mathbf{H}_2$ synthesis are artifacts of Lagrange duality in optimization problems whose variables are Lyapunov functions.

7.3 Current and future directions

Terascale networks. Recent years have seen an explosion of interest in large scale sensing and actuation networks, as evidenced by emerging applications related to the smart city, transportation engineering, building planning, and power grid optimization. Such applications are enabled by increasingly cheaper, smaller, and more powerful sensors and computational devices that can be connected together in a network.

The challenge facing engineers of such systems is twofold. First, such systems are usually decentralized, because no single agent in the network can be expected to have access to all data relevant to making a global inference. Second, these devices are usually limited by power, which translates to restrictions on computation and communication. However, there is strength in numbers. What cannot be done by one device because of computation or communication constraints can sometimes be done by many devices collaborating on a single goal.

The distributed system identification framework proposed in this chapter serves to take advantage of large numbers of intelligent systems to trade off time (of computation, of inherent dynamics), and hence, of power, against space by adding more devices to the network. In this sense, simple rules like eq. (6.3) can easily be implemented by devices that are limited by computation and communication, leading to a global collaborative behavior.

For example, θ can very well describe a global environment state or a map. If each agent is confined to a particular area of a map, and can measure a correlation between θ and some regressor ϕ_i, then the information sharing framework, suitably adapted to a discrete setting of, *e.g.*, an occupancy grid, as well as discrete time, allows for multiple agents to collaboratively identify the entire space without any single agent having to explore every part of the space. The defining conveniences of such schemes are several:

- *Sensing locality*: each agent only directly operates on local sensor information, or information available by direct communication.

- *Communication*: collaborative identification is entirely decentralized, in that only those agents allowed to communicate with each other do so, whether it be due to power, channel capacity, or geographical constraints.

- *Real-time*: governed by the parameters of the network itself, each agent achieves

asymptotic (or, in the case of discrete-time schemes on classes of communication graphs termed "expander" graphs, in finite time) decision accuracy.

- *Graceful degradation and adaptation*: if the environment state θ changes, networked adaptive systems are able to detect these changes and adapt their decisions.

- *Robustness*: the network provides a source of robustness for the decision rules. If an agent fails and can detect the failure, it can still achieve the same level of decision accuracy by simply listening to the network.

Machine learning. With increased sensing capability comes an increase in the amount of data. Effectively dealing with "Big Data" poses computational and inferential challenges in modern systems. Ideas from consensus and adaptation can be applied in this setting as well. In this case, θ is a set of modeling parameters upon which inferential hypotheses may be tested. The machine learning goal can be stated as to gather as much data (through the values of $\phi(x_i)$ and y_i) as possible, and determine a summary θ of the data useful for the inference task. In the extreme case, every agent i corresponds to a data point, and schemes like eq. (6.3) represent rules for updating parameter estimates as data become available. Consensus on θ is a way to enforce the prior belief that all data points originate from the same distribution.

Human factors. Control systems, such as the one depicted in Figure 7.1, are typically specified in terms of three sometimes competing goals:

1. *robust stability goal*: $\nabla V(x)^T f(x,u) < 0$, $\forall$plants
2. *performance goal*: $\nabla V(x)^T f(x,u) < -\ell(x,u)$, $\forall$plants
3. *learning goal*: $\int_t^{t+T} \phi\phi^T \, dt \succ 0$, $\forall t$

While the purview of control theory is usually in designing an automatic "controller" (linear state feedback, MPC, ADP...) to satisfy the given goals, the language of Lyapunov bounds can equally well be applied to a human in the loop controller. Because people are often much better at making complicated, nonconvex decisions than computers, it is not too difficult to envision future systems in which a human decision making strategy is augmented by computed Lyapunov bounds that guarantee a certain level of closed loop stability and

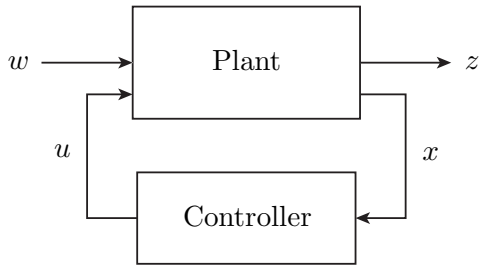


Figure 7.1: People as control systems.

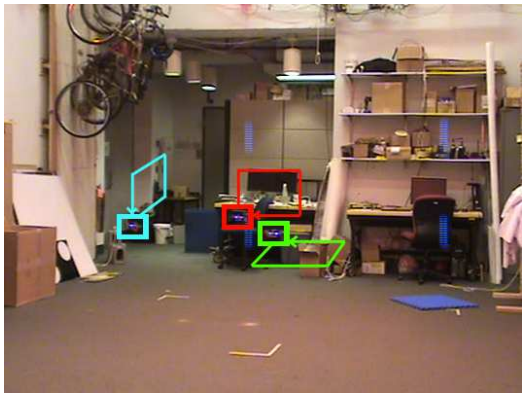
robustness. As long as a decision maker ensures the relevant robustness metrics are obeyed when making a decision, system theoretic stability is guaranteed. Meanwhile, things that computers are typically bad at, such as exploration and learning, can be offloaded to people, and gauged by Lyapunov bounds.

Sampling based strategies for ADP and automatic tools. Once under- and over-approximations of the Lyapunov functions are given as semiinfinite constraints parameterized by the states and inputs, constraint sampling and active set techniques can be used to perform approximate optimization over these functional sets. Extensions to nonlinear systems and more complicated hybrid dynamics can be made. Moreover, in the case of co-safe LTL constrained hybrid systems, co-construction of the optimization problem and the specification automaton can allow for structure exploitation and the ability to consider medium to large dimensional spaces without relying on state space discretization.

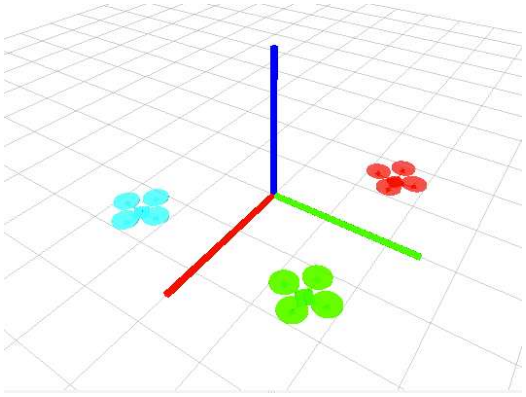
Circuits. Ideas from networked systems can be used to inform classically "centralized" applications, such as detecting the phase of a signal in a phase-locked loop (PLL). Here, the parameter θ refers to the phase of an incoming signal, which may change on a slower timescale than the convergence of the update dynamics (6.3), and each agent corresponds to a different PLL instance, placed in different locations on a single chip die.

The consensus dynamics (6.3) can in fact be implemented in physical circuitry (*e.g.*, resistors and capacitors), rather than sampled in a digital manner as would be the case with macroscopic agents in terascale networks. Information fusion and voting can be implemented at the lowest practical level, and would be limited by fundamental physical parameters, such as the aspect ratios of communication buses, thermal noise, and the areas of amplifier input stages at the input of each PLL.

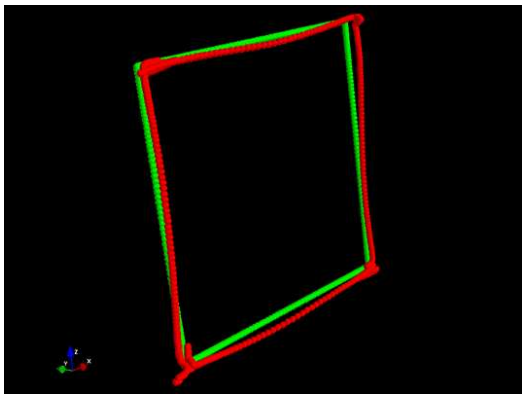
Micro UAV formation scale extraction. Formations of dynamic agents have been studied in parallel with consensus, and at this point the theory of pure linear and in some cases nonlinear consensus is well developed. However, applications are still lacking. In particular, what happens when a consensus scheme is used to make decisions on the scale of a formation of UAVs while the scale itself is being estimated? Exploratory work in applying the networked adaptive systems idea (See Figure 7.2) has been done by the author during a weeklong visit to the GRASP lab at University of Pennsylvania.



(a) Experimental setup



(b) Simulator setup



(c) Trajectory visualization

Figure 7.2: (a) Three "nano" quadrotors (boxed) in simultaneous flight and an assigned plane of motion for each (indicated by arrow) used to distribute a system identification task among the three agents. (b) High fidelity simulation of quadrotor dynamics corresponding to the experimental setup. (c) Visualization of desired and executed motions.

Appendix A

Proof of Theorem 4 for $p > 1$

Form column vectors $\hat{\theta} = (\hat{\theta}_1, \ldots, \hat{\theta}_n) \in \mathbf{R}^{np}$ and $\tilde{\theta} = (\tilde{\theta}_1, \ldots, \tilde{\theta}_n) \in \mathbf{R}^{np}$ by stacking the components $\hat{\theta}_i \in \mathbf{R}^p$ and $\tilde{\theta}_i = \hat{\theta}_i - \theta \in \mathbf{R}^p$ for all $i = 1, \ldots, n$. The dynamics (6.9) are now

$$\frac{d}{dt}\tilde{\theta} = -(L \otimes I_p)\tilde{\theta} - \gamma\Phi(t)\tilde{\theta},$$

where $\otimes$ is the Kronecker product, $I_p \in \mathbf{R}^{p\times p}$ is the identity matrix, and $\Phi : [0, \infty) \to \mathbf{R}^{np\times np}$ is block diagonal,

$$\Phi(t) = \begin{bmatrix} \phi_1(t)\phi_1(t)^T & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi_n(t)\phi_n(t)^T \end{bmatrix}.$$

The candidate Lyapunov function

$$V(\tilde{\theta}) = \frac{1}{2}\tilde{\theta}^T\tilde{\theta} = \frac{1}{2}\sum_{i=1}^{n}\tilde{\theta}_i^T\tilde{\theta}_i$$

has nonpositive derivative

$$\dot{V}(\tilde{\theta}) = -\tilde{\theta}^T\big((L \otimes I_p) + \gamma\Phi(t)\big)\tilde{\theta} \leq 0,$$

with $\dot{V} \to 0$ as $t \to \infty$ by the same arguments as before, thus parts 1-3 follow. For part 4, the mixed product property $AB \otimes CD = (A \otimes C)(B \otimes D)$ for appropriately sized matrices A, B, C, and D implies that the spectrum of $L \otimes I_p$ is related to the spectrum of L and I_p

by

$$(L \otimes I_p)\left(\frac{1}{\sqrt{n}}\mathbf{1} \otimes e_j\right) = 0,$$
$$(L \otimes I_p)(v_i \otimes e_j) = \lambda_i (v_i \otimes e_j),$$

for all $i = 2, \ldots, n$ and $j = 1, \ldots, p$, where $e_j \in \mathbf{R}^p$ is the jth unit vector. Write a unit vector $w \in \mathbf{R}^{np}$ in this basis as

$$w = \sum_{j=1}^{p} \alpha_j \frac{1}{\sqrt{n}}\mathbf{1} \otimes e_j + \sum_{i=2}^{n}\sum_{j=1}^{p} \beta_{ij} v_i \otimes e_j,$$

with $(\alpha, \beta) \in \mathbf{R}^p \times \mathbf{R}^{(n-1)p}$ having unit norm. As before, let $\bar{\Phi}$ be the average of Φ over $[t_0, t]$. We wish to bound

$$\max\{w^T(L \otimes I_p)w, \gamma w^T \bar{\Phi} w\}$$

uniformly below by a strictly positive constant. Using the mixed product property and $\|\alpha\|_2^2 + \|\beta\|_2^2 = 1$ we have

$$\begin{aligned} w^T(L \otimes I_p)w &= \sum_{i=2}^{n}\sum_{j=1}^{p} \lambda_i \beta_{ij}^2 \\ &\geq \lambda_2(1 - \|\alpha\|_2^2). \end{aligned}$$

For the second term $w^T \bar{\Phi} w$, note that

$$(\mathbf{1} \otimes e_i)^T \bar{\Phi}(\mathbf{1} \otimes e_j) = (\overline{\phi_1 \phi_1^T})_{ij} + \cdots + (\overline{\phi_n \phi_n^T})_{ij},$$

hence

$$
\begin{aligned}
w^T\bar{\Phi}w &= \frac{1}{n}\sum_{i=1}^{p}\sum_{j=1}^{p}\alpha_i\alpha_j(\mathbf{1}\otimes e_i)\bar{\Phi}(\mathbf{1}\otimes e_j) \\
&\quad + \frac{2}{\sqrt{n}}\sum_{i=2}^{n}\sum_{j=1}^{p}\sum_{k=1}^{p}\alpha_k\beta_{ij}(\mathbf{1}\otimes e_k)^T\bar{\Phi}(v_i\otimes e_j) \\
&\quad + \underbrace{\sum_{i=2}^{n}\sum_{j=1}^{p}\sum_{k=2}^{n}\sum_{l=1}^{p}\beta_{ij}\beta_{kl}(v_i\otimes e_j)\bar{\Phi}(v_k\otimes e_l)}_{\geq 0} \\
&\geq \frac{1}{n}\alpha^T\left(\sum_{i=1}^{n}\overline{\phi_i\phi_i^T}\right)\alpha \\
&\quad - \frac{2m_2 n}{\sqrt{n}}\sum_{i=2}^{n}\sum_{j=1}^{p}\sum_{k=1}^{p}|\alpha_k\beta_{ij}| \\
&\geq \frac{\|\alpha\|_2^2}{n}m_1 - 2m_2 n\sqrt{\|\alpha\|_2^2(1-\|\alpha\|_2^2)},
\end{aligned}
$$

thus a loose uniform lower bound is

$$\max\{w^T(L\otimes I_p)w, \gamma w^T\bar{\Phi}w\} \geq a > 0.$$

A continuity argument should convince the reader that

$$
\begin{aligned}
a = \inf_{\|\alpha\|_2\leq 1}\max\Big\{&\lambda_2(1-\|\alpha\|_2^2), \\
&\gamma\frac{\|\alpha\|_2^2}{n}m_1 - 2\gamma m_2 n\sqrt{\|\alpha\|_2^2(1-\|\alpha\|_2^2)}\Big\}
\end{aligned}
$$

is strictly positive.

Bibliography

[AHLP00] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, July 2000.

[And77] B. D. O. Anderson. Exponential stability of linear equations arising in adaptive identification. *IEEE Transactions on Automatic Control*, 22(1):83–88, 1977.

[And05] B. D. O. Anderson. Failures of adaptive control theory and their resolution. *Communications in Information & Systems*, 5(1):1–20, 2005.

[BEFB94] S. P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied and Numerical Mathematics*. Society for Industrial and Applied Mathematics, 1994.

[Bel57] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[Ber76] D. P. Bertsekas. Multiplier methods: A survey. *Automatica*, 12(2):133–145, March 1976.

[Ber96] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[Ber05] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume I and II. Athena Scientific, 3rd edition, 2005.

[BK08] C. Baier and J.-P. Katoen. *Principles of Model Checking*. Representation and Mind. MIT Press, 2008.

[BM99] A. Bemporad and M. Morari. *Robustness in Identification and Control*, chapter Robust Model Predictive Control: a Survey, pp. 207–226. Springer, London, 1999.

[BMKV11] A. Bhatia, M. R. Maly, L. E. Kavraki, and M. Y. Vardi. Motion planning with complex goals. *IEEE Robotics Automation Magazine*, 18(3):55–64, September 2011.

[Boy08] S. P. Boyd. EE363 course notes, 2008. Available at `http://stanford.edu/class/ee363/`.

[BPC+11] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[BPG89] J. Bernussou, P. Peres, and J. Geromel. A linear programming oriented procedure for quadratic stabilization of uncertain systems. *Systems & Control Letters*, 13(1):65–72, 1989.

[BS83] S. P. Boyd and S. Sastry. On parameter convergence in adaptive control. *Systems & Control Letters*, 3(6):311–319, December 1983.

[BT96] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[BV04] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[CFH+03] E. Clarke, A. Fehnker, Z. Han, B. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *International Journal of Foundations of Computer Science*, 14(04):583–604, 2003.

[CHIK03] R. R. Costa, L. Hsu, A. K. Imai, and P. Kokotović. Lyapunov-based adaptive control of MIMO systems. *Automatica*, 39(7):1251–1257, 2003.

[CL55] E. A. Coddington and N. Levinson. *Theory of Ordinary Differential Equations*. McGraw–Hill, New York, 1955.

[Cor08] J. Cortés. Discontinuous dynamical systems. *IEEE Control Systems*, 28(3):36–73, June 2008.

[Dav05] T. A. Davis. Algorithm 849: A concise sparse Cholesky factorization package. *ACM Transactions on Mathematical Software*, 31(4):587–591, December 2005.

[Dav06] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2006.

[dFR03] D. P. de Farias and B. V. Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[DGKF89] J. C. Doyle, K. Glover, P. P. Khargonekar, and B. A. Francis. State-space solutions to standard $\mathbf{H}_2$ and $\mathbf{H}_\infty$ control problems. *IEEE Transactions on Automatic Control*, 34(8):831–847, August 1989.

[DJ14] A. Domahidi and J. Jerez. FORCES Professional. embotech GmbH (`http://embotech.com/FORCES-Pro`), July 2014.

[DP00] G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory: A Convex Approach*. Texts in Applied Mathematics. Springer, 2000.

[DS81] J. C. Doyle and G. Stein. Multivariable feedback design: Concepts for a classical/modern synthesis. *IEEE Transactions on Automatic Control*, pp. 4–16, 1981.

[Duf04] I. S. Duff. MA57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Transactions on Mathematical Software*, 30(2):118–144, June 2004.

[Dyd10] Z. T. Dydek. *Adaptive control of Unmanned Aerial Systems*. Ph.D. thesis, Massachusetts Institute of Technology, 2010.

[ERA05] C. Ebenbauer, J. Renz, and F. Allgöwer. Polynomial feedback and observer design using nonquadratic Lyapunov functions. In *IEEE Conference on Decision and Control and the European Control Conference (CDC-ECC)*, pp. 7587–7592. December 2005.

[FGKGP09] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[FK96] R. A. Freeman and P. V. Kokotović. *Robust Nonlinear Control Design: State-Space and Lyapunov Techniques*. Systems & Control: Foundations & Applications. Birkhäuser, 1996.

[FPMM16] S. S. Farahani, I. Papusha, C. McGhan, and R. M. Murray. Constrained autonomous satellite docking via differential flatness and model predictive control. In *IEEE Conference on Decision and Control (CDC), submitted*. 2016.

[FT13] J. Fu and H. G. Tanner. Bottom-up symbolic control: Attractor-based planning and behavior synthesis. *IEEE Transactions on Automatic Control*, 58(12):3142–3155, 2013.

[GB08] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pp. 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[GB14] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[GD13] M. Guo and D. V. Dimarogonas. Nonlinear consensus via continuous, sampled, and aperiodic updates. *International Journal of Control*, 86(4):567–578, 2013.

[GO01] P. Gastin and D. Oddoux. Fast LTL to Büchi automata translation. In G. Berry, H. Comon, and A. Finkel, editors, *International Conference*

on Computer Aided Verification (CAV'01), volume 2102 of *Lecture Notes in Computer Science*, pp. 53–65. Springer, Paris, France, July 2001.

[HB10] L. C. G. J. M. Habets and C. Belta. Temporal logic control for piecewise-affine hybrid systems on polytopes. In *International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, pp. 195–202. July 2010.

[HPB14] M. B. Horowitz, I. Papusha, and J. W. Burdick. Domain decomposition for stochastic optimal control. In *IEEE Conference on Decision and Control (CDC)*, pp. 1866–1873. 2014.

[HR99] S. Hedlund and A. Rantzer. Optimal control of hybrid systems. In *IEEE Conference on Decision and Control (CDC)*, volume 4, pp. 3972–3977. 1999.

[HR02] S. Hedlund and A. Rantzer. Convex dynamic programming for hybrid systems. *IEEE Transactions on Automatic Control*, 47(9):1536–1540, 2002.

[IF06] P. Ioannou and B. Fedan. *Adaptive Control Tutorial*. Advances in Design and Control. Society for Industrial and Applied Mathematics, 2006.

[IK83] P. A. Ioannou and P. V. Kokotović. *Adaptive Systems with Reduced Models*. Springer, 1983.

[JR98] M. Johansson and A. Rantzer. Computation of piecewise quadratic Lyapunov functions for hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):555–559, April 1998.

[Kak03] S. M. Kakade. *On the Sample Complexity of Reinforcement Learning*. Ph.D. thesis, University College London, March 2003.

[KB08] M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Transactions on Automatic Control*, 53(1):287–297, 2008.

[KBM96] M. V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained Model Predictive Control using Linear Matrix Inequalities. *Automatica*, 32(10):1361–1379, 1996.

[KF12] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning with deterministic μ-calculus specifications. In *American Control Conference (ACC)*, pp. 735–742. June 2012.

[Kha02] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 3rd edition, 2002.

[KN82] G. Kreisselmeier and K. S. Narendra. Stable model reference adaptive control in the presence of bounded disturbances. *IEEE Transactions on Automatic Control*, 27(6):1169–1175, December 1982.

[Kra63] N. N. Krasovskii. *Stability of Motion*. Stanford University Press, 1963.

[KSS+13] N. Kariotoglou, S. Summers, T. Summers, M. Kamgarpour, and J. Lygeros. Approximate dynamic programming for stochastic reachability. In *European Control Conference (ECC)*, pp. 584–589. July 2013.

[Kur55a] J. Kurzweil. On the reversibility of the first theorem of Lyapunov concerning the stability of motion. *Czechoslovak Mathematical Journal*, 5(3):382–398, 1955. In Russian.

[Kur55b] J. Kurzweil. On the reversibility of the second theorem of Lyapunov concerning the stability of motion. *Czechoslovak Mathematical Journal*, 5(3):435–438, 1955. In Russian.

[KV01] O. Kupferman and M. Y. Vardi. Model checking of safety properties. *Formal Methods in System Design*, 19(3):291–314, November 2001.

[Lö4] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *CACSD Conference*. Taipei, Taiwan, 2004.

[LH07] E. Lavretsky and N. Hovakimyan. Stable adaptation in the presence of input constraints. *Systems & Control Letters*, 56(11–12):722–729, 2007.

[LN88] T.-H. Lee and K. S. Narendra. Robust adaptive control of discrete-time systems using persistent excitation. *Automatica*, 24(6):781–788, 1988.

[LO14] J. Liu and N. Ozay. Abstraction, discretization, and robustness in temporal logic control of dynamical systems. In *International Conference on Hybrid Systems: Computation and Control (HSCC)*, pp. 293–302. ACM, 2014.

[LSW96] Y. Lin, E. D. Sontag, and Y. Wang. A smooth converse Lyapunov theorem for robust stability. *SIAM Journal on Control and Optimization*, 34(1):124–160, 1996.

[LW13] E. Lavretsky and K. A. Wise. *Robust and Adaptive Control: with Aerospace Applications*. Advanced Textbooks in Control and Signal Processing. Springer, 2013.

[Lya92] A. M. Lyapunov. The general problem of the stability of motion. 1892. In Russian.

[Lya92] A. M. Lyapunov. The general problem of the stability of motion. *International Journal of Control*, 55(3):531–773, 1992. Translated into English by A. T. Fuller.

[Mas56] J. L. Massera. Contributions to stability theory. *Annals of Mathematics*, 64(1):182–206, 1956.

[MB11] J. Mattingley and S. P. Boyd. CVXGEN: a code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, 2011.

[MN77] A. P. Morgan and K. S. Narendra. On the uniform asymptotic stability of certain linear nonautonomous differential equations. *SIAM Journal on Control and Optimization*, 15(1):5–24, 1977.

[MP92] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag, 1992.

[MR97] A. Megretski and A. Rantzer. System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, 42(6):819–830, June 1997.

[MWB11] J. Mattingley, Y. Wang, and S. P. Boyd. Receding horizon control: Automatic generation of high-speed solvers. *IEEE Control Systems Magazine*, 31(3):52–65, June 2011.

[NA86] K. S. Narendra and A. M. Annaswamy. Robust adaptive control in the presence of bounded disturbances. *IEEE Transactions on Automatic Control*, AC-31(4):306–315, April 1986.

[NA87] K. S. Narendra and A. M. Annaswamy. A new adaptive law for robust adaptation without persistent excitation. *IEEE Transactions on Automatic Control*, AC-32(2):134–145, February 1987.

[NA05] K. S. Narendra and A. M. Annaswamy. *Stable Adaptive Systems*. Dover Publications, 2005.

[Nes04] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Applied Optimization. Springer, 2004.

[NN94] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization*. Operations Research and Financial Engineering. Springer, 2nd edition, 2006.

[OSFM07] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[OSM03] R. Olfati-Saber and R. M. Murray. Consensus protocols for networks of dynamic agents. In *American Control Conference (ACC)*, volume 2, pp. 951–956. 2003.

[OSM04] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[Par00] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. Ph.D. thesis, California Institute of Technology, 2000.

[PD93] A. Packard and J. C. Doyle. The complex structured singular value. *Automatica*, 29(1):71–109, 1993.

[Per01] L. Perko. *Differential Equations and Dynamical Systems*. Springer, New York, 3 edition, 2001.

[PFTM16] I. Papusha, J. Fu, U. Topcu, and R. M. Murray. Automata theory meets approximate dynamic programming: Optimal control with temporal logic constraints. In *IEEE Conference on Decision and Control (CDC), submitted*. 2016.

[PKP09] D. Peaucelle, H. M. Khan, and P. V. Pakshin. LMI-based analysis of robust adaptive control for linear systems with time-varying uncertainty. *Automation and Remote Control*, 70(9):1540–1552, 2009.

[PLM14] I. Papusha, E. Lavretsky, and R. M. Murray. Collaborative system identification via parameter consensus. In *American Control Conference (ACC)*, pp. 13–19. June 2014.

[PM15] I. Papusha and R. M. Murray. Analysis of control systems on symmetric cones. In *IEEE Conference on Decision and Control (CDC)*, pp. 3971–3976. December 2015.

[PN82] B. B. Peterson and K. S. Narendra. Bounded error adaptive control. *IEEE Transactions on Automatic Control*, AC-27(6):1161–1982, 1982.

[Pow07] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics. Wiley-Interscience, 2007.

[PP92] J.-B. Pomet and L. Praly. Adaptive nonlinear regulation: Estimation from the Lyapunov equation. *IEEE Transactions on Automatic Control*, 37(6):729–740, June 1992.

[PS82] Y. S. Pyatnitskiy and V. I. Skorodinskiy. Numerical methods of Lyapunov function construction and their application to the absolute stability problem. *Systems & Control Letters*, 2(2):130–135, 1982.

[PS91] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.

[Ran96] A. Rantzer. On the Kalman–Yakubovich–Popov lemma. *Systems & Control Letters*, 28(1):7–10, 1996.

[Ran99] A. Rantzer. Dynamic programming via convex optimization. In *IFAC World Congress*, pp. 491–496. 1999.

[Rei11] G. Reissig. Computing abstractions of nonlinear systems. *IEEE Transactions on Automatic Control*, 56(11):2583–2598, 2011.

[Ren08] W. Ren. On consensus algorithms for double-integrator dynamics. *IEEE Transactions on Automatic Control*, 53(6):1503–1509, 2008.

[RL06] M. Rotkowitz and S. Lall. A characterization of convex problems in decentralized control. *IEEE Transactions on Automatic Control*, 51(2):274–286, February 2006.

[RM09] J. B. Rawlings and D. Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2009.

[SB89] S. Sastry and M. Bodson. *Adaptive Control: Stability, Convergence and Robustness*. Prentice Hall, 1989.

[SKK+13] T. H. Summers, K. Kunz, N. Kariotoglou, M. Kamgarpour, S. Summers, and J. Lygeros. Approximate dynamic programming via sum of squares programming. In *European Control Conference (ECC)*, pp. 191–197. July 2013.

[Son98] E. D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. Textbooks in Applied Mathematics. Springer, New York, 2 edition, 1998.

[SRS09] M. Schwager, D. Rus, and J.-J. Slotine. Decentralized, adaptive coverage control for networked robots. *International Journal of Robotics Research*, 28(3):357–375, March 2009.

[SSS11] S. S. Stanković, M. S. Stanković, and D. M. Stipanović. Decentralized parameter estimation by consensus based stochastic approximation. *IEEE Transactions on Automatic Control*, 56(3):531–543, 2011.

[STBR11] S. L. Smith, J. Tůmova, C. Belta, and D. Rus. Optimal path planning for surveilance with temporal-logic constraints. *International Journal of Robotics Research*, 30(14):1695–1708, December 2011.

[Ste03] G. Stein. Respect the unstable. *IEEE Control Systems*, 23(4):12–25, August 2003.

[Stu99] J. F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[TTT99] K.-C. Toh, M. J. Todd, and R. H. Tutuncu. SDPT3—a MATLAB software package for semidefinite programming, 1999.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

[Van10] L. Vandenberghe. The CVXOPT linear and quadratic cone program solvers. `http://www.seas.ucla.edu/~vandenbe/publications/coneprog.pdf`, 2010.

[WB09] Y. Wang and S. P. Boyd. Performance bounds for linear stochastic control. *Systems & Control Letters*, 58(3):178–182, 2009.

[Wil72a] J. C. Willems. Dissipative dynamical systems part I: General theory. *Archive for Rational Mechanics and Analysis*, 45(5):321–351, 1972.

[Wil72b] J. C. Willems. Dissipative dynamical systems part II: Linear systems with quadratic supply rates. *Archive for Rational Mechanics and Analysis*, 45(5):352–393, 1972.

[Wil07] J. C. Willems. In control, almost from the beginning until the day after tomorrow. *European Journal of Control*, 13(1):71–81, 2007. Special issue: On the Dawn and Development of Control Science in the XXth Century.

[WM13] E. M. Wolff and R. M. Murray. Optimal control of nonlinear systems with temporal logic specifications. In *International Symposium on Robotics Research (ISRR)*. 2013.

[WOB14] Y. Wang, B. O'Donoghue, and S. P. Boyd. Approximate dynamic programming via iterated Bellman inequalities. *International Journal of Robust and Nonlinear Control*, 2014.

[WS06] W. Wang and J.-J. Slotine. A theoretical study of different leader roles in networks. *IEEE Transactions on Automatic Control*, 51(7):1156–1161, 2006.

[WTM12] T. Wongpiromsarn, U. Topcu, and R. M. Murray. Receding horizon temporal logic planning. *IEEE Transactions on Automatic Control*, 57(11):2817–2830, November 2012.

[WTM13] E. M. Wolff, U. Topcu, and R. M. Murray. Automaton-guided controller synthesis for nonlinear systems with temporal logic. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4332–4339. November 2013.

[WZL03] M. Wu, L. Zhang, and G. Liu. Adaptive robust $\mathbf{H}_\infty$ control for nonlinear systems with parametric uncertainties and external disturbances. In *European Control Conference (ECC)*, pp. 826–832. September 2003.

[XA04] X. Xu and P. J. Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):2–16, 2004.

[XB04] L. Xiao and S. P. Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53:65–78, February 2004.

[XBL05] L. Xiao, S. P. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 63–70. 2005.

[Yak62] V. A. Yakubovich. The solution of certain matrix inequalities in automatic control theory. *Doklady Akademii Nauk SSSR*, 143(6):1304–1307, 1962.

[Yak92] V. A. Yakubovich. Nonconvex optimization problem: The infinite-horizon linear-quadratic control problem with quadratic constraints. *Systems & Control Letters*, 19(1):13–22, 1992.

[YCCK10] W. Yu, G. Chen, M. Cao, and J. Kurths. Second-order consensus for multiagent systems with directed topologies and nonlinear dynamics. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 40(3):881–891, June 2010.

[YW77] J. S.-C. Yuan and W. M. Wonham. Probing signals for model reference identification. *IEEE Transactions on Automatic Control*, 22(4):530–538, 1977.

[YYCS09] B.-J. Yang, T. Yucelen, A. Calise, and J.-Y. Shin. LMI-based analysis for stability margins of adaptive flight control. In *AIAA Guidance, Navigation, and Control Conference*, pp. AIAA 2009–6189. August 2009.

[Zam81] G. Zames. *IEEE Transactions on Automatic Control*, 26(2):301–320, April 1981.